NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE APPLICATION OF A VIEWPOINTS FRAMEWORK IN THE DEVELOPMENT OF C4I SYSTEMS

by

Sheila A. Smith

June 2000

Thesis Advisor:	James Bret Michael
Co-Advisor:	William G. Kemple

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. **AGENCY USE ONLY** *(Leave blank)*	2. **REPORT DATE** June 2000	3. **REPORT TYPE AND DATES COVERED** Master's Thesis

4. **TITLE AND SUBTITLE:** The Application of a Viewpoints Framework in the Development of C4I Systems	5. **FUNDING NUMBERS**
6. **AUTHOR(S)** Smith, Sheila A.	
7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School Monterey, CA 93943-5000	8. **PERFORMING ORGANIZATION REPORT NUMBER**
9. **SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**	10. **SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. **SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. **DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited.	12b. **DISTRIBUTION CODE**

13. **ABSTRACT** *(maximum 200 words)*

In the development of large distributed systems, both the detection and resolution of inconsistency in policy, requirements, and specifications pose major challenges. The purpose of this thesis is to examine the inconsistencies in policy, requirements, and specifications in the development of information/Joint Command, Control, Communications, Computers, and Intelligence (C4I) systems. In this thesis, we explore the application of a "viewpoints" framework to aid in the development of distributed information systems.

A viewpoints framework methodology that was developed to aid in the development of distributed systems is the Reference Model of Open Distributed Processing (RM-ODP). This thesis is concerned with the application of the five viewpoints of RM-ODP and the translation of policy into requirements and specifications. In this thesis we use the Ballistic Missile Defense (BMD) system as a case study to explain how RM-ODP can be used to develop distributed information systems. We found that identifying inconsistencies regarding interoperability amongst the subsystems of BMD necessitated the use of multiple viewpoints and that firm conclusions could not be made until the system was viewed at the lower levels.

14. **SUBJECT TERMS** Ballistic Missile Defense,' C4I, Interoperability, Policy, Reference Model of Open Distributed Processing, , Requirements Engineering, Viewpoints	15. **NUMBER OF PAGES** 100
	16. **PRICE CODE**

17. **SECURITY CLASSIFICATION OF REPORT** Unclassified	18. **SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	19. **SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	20. **LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THE APPLICATION OF A VIEWPOINTS FRAMEWORK IN THE DEVELOPMENT OF C4I SYSTEMS

Sheila A. Smith
Lieutenant, United States Navy
B.S., Illinois State University, 1994

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN COMPUTER SCIENCE
AND
MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY

from the

NAVAL POSTGRADUATE SCHOOL
June 2000

Author: Sheila A. Smith
Sheila A. Smith

Approved by: James Bret Michael
James Bret Michael, Thesis Advisor

[illegible]
William G. Kemple, Thesis Co-Advisor

Dan C Boger
Dan C. Boger, Chairman
Computer Science Department and C3 Academic Group

ABSTRACT

In the development of large distributed systems, both the detection and resolution of inconsistency in policy, requirements, and specifications pose major challenges. The purpose of this thesis is to examine the inconsistencies in policy, requirements, and specifications in the development of information/Joint Command, Control, Communications, Computers, and Intelligence (C4I) systems. In this thesis, we explore the application of a "viewpoints" framework to aid in the development of distributed information systems.

A viewpoints framework methodology that was developed to aid in the development of distributed systems is the Reference Model of Open Distributed Processing (RM-ODP). This thesis is concerned with the application of the five viewpoints of RM-ODP and the translation of policy into requirements and specifications. In this thesis we use the Ballistic Missile Defense (BMD) system as a case study to explain how RM-ODP can be used to develop distributed information systems. We found that identifying inconsistencies regarding interoperability amongst the subsystems of BMD necessitated the use of multiple viewpoints and that firm conclusions could not be made until the system was viewed at the lower levels.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

In the development of large distributed information systems, both the detection and resolution of inconsistency in policy, requirements, and specifications pose major challenges. The purpose of this thesis is to examine inconsistencies in policy, requirements, and specifications in the development of distributed information/Joint Command, Control, Communications, Computers, and Intelligence (C4I) systems. In this thesis, we explore the application of a "viewpoints" framework to aid in the development of distributed information systems.

A viewpoints framework methodology that was developed to aid in the development of distributed systems is the Reference Model of Open Distributed Processing (RM-ODP). This thesis is concerned with the application of the five viewpoints of RM-ODP and the translation of policy into requirements and specifications. A case study of the Ballistic Missile Defense (BMD) system is presented in order to illustrate how RM-ODP can be used in the definition of requirements and specifications and in the detection of inconsistencies in policy, requirements, and specifications. The BMD system is a good candidate to model using viewpoints because it is a complex, distributed, heterogeneous system comprised of many subsystems.

Ballistic missile defense involves many challenges. One of these challenges is the ability to hit a moving target. Another major challenge facing BMD is that of interoperability amongst all the systems. The RM-ODP viewpoints can be used to aid in modeling the BMD system to help address some of these challenges. Through the use of

RM-ODP, the various members of the development team can communicate their policy, requiements, and specifications for the composite system using a common set of languages and mappings between the languages (i.e, levels in the five-layer viewpoint model).

In the case study the interoperability problem facing the BMD system is addressed using the RM-ODP viewpoints framework. The case study demonstrates how the viewpoints can be applied at a very abstract level.

Much research has been done in the area of viewpoints. The concepts of viewpoints and inconsistency management in software development are introduced. Much of the previous research stresses the importance of conflicts in software development and suggests that conflicts are a necessary part of the development process and should be represented and understood. The use of a policy workbench to aid in analyzing proposed policy, maintaining a policy database, and assisting in policy enforcement is also discussed. Lastly, policy conflicts and the problems of conflict detection and resolution in distributed systems are presented.

We found that identifying inconsistencies regarding interoperability amongst the subsystems of BMD necessitated the use of multiple viewpoints and that firm conclusions could not be made until the system was viewed at the lower levels. These findings can be used by U.S. DoD as a tool to aid in the development of distributed information systems.

I. INTRODUCTION

A. PROBLEM STATEMENT

Many questions remain to be answered regarding how to best develop large distributed systems. Among other things, both the detection and resolution of inconsistencies in policy, requirements, and specifications pose major challenges.

A distributed system is a set of computers, connected by at least one network, that do not share memory or a common clock. The goal of a distributed system is to cause a set of computers to appear to the user as a single, powerful virtual machine. There are five primary advantages offered by distributed systems: resource sharing (hardware and software can be shared among computers); enhanced performance (higher throughput and speed through increased concurrency); improved reliability (through replication of data files and services, distributed systems can be made more fault tolerant); improved availability (some components may fail without affecting the overall availability of the system); and modular expandability (hardware and software can be added without adversely impacting existing resources).

The purpose of this thesis is to examine the inconsistencies in policy (rules or a means of specifying and influencing behavior within a distributed system), requirements, and specifications in the development of distributed information/Joint Command, Control, Communications, Computers, and Intelligence (C4I) systems. In this thesis, we explore the application of a "viewpoints" framework to aid in the development of distributed information systems. A viewpoints framework is used to partition a system specification into a number of different components. Each component (viewpoint) is a complete and self-contained description of the required distributed system targeted towards a particular audience (development team participant).

B. MOTIVATION

One viewpoints framework methodology that was developed to aid in the development of distributed systems is the Reference Model of Open Distributed Processing (RM-ODP). RM-ODP allows developers to view the system as an interrelated whole rather than as a collection of parts, and provides a way in which distributed systems and their features can be described. Much work has been done with the RM-ODP approach to systems development. This thesis is concerned with the application of the five viewpoints of RM-ODP and the translation of policy into requirements and specifications. This thesis will use the Ballistic Missile Defense system (which is a large distributed system) as a case study to explain how RM-ODP can be used in the definition of requirements and specifications and in the detection of inconsistencies in policy, requirements, and specifications.

C. SCOPE

The scope of this thesis is limited to application of the five viewpoints of RM-ODP to managing inconsistency when developing distributed information systems, in particular the Ballistic Missile Defense system. This research will provide conclusions and recommendations for dealing with inconsistencies in the definition of policy, requirements, and specifications for information systems/Joint C4I systems. The resulting conclusions and recommendations in this research are intended to aid the U.S. Department of Defense (DoD) in the development of systems so that inconsistencies are caught as early in the life cycle as possible.

D. ORGANIZATION

The organization of this thesis is as follows: Chapter II provides an overview of RM-ODP and discusses the five viewpoints in detail. Chapter III describes some of the previous research most closely related to this thesis. The concepts of viewpoints and inconsistency management in software development are introduced. Much of the previous research stresses the importance of conflicts in software development and suggests that conflicts are a necessary part of the development process and should be represented and understood. The use of a policy workbench to aid in analyzing proposed policy, maintain a policy database, and assist in policy enforcement is also discussed. Lastly, policy conflicts and the problems of conflict detection and resolution in distributed systems are presented. Chapter IV provides an overview of the BMD system. Chapter V uses the BMD system as case study to demonstrate how RM-ODP can be used to aid in the definition of requirements and specifications. Chapter VI provides the resulting conclusions and recommendations in this research. Suggestions for future research are also discussed.

THIS PAGE INTENTIONALLY LEFT BLANK

II. RM-ODP

A. INTRODUCTION

Advances in computer networking have allowed computer systems across the world to be interconnected. Despite this, heterogeneity of computer systems is still a problem. Open Distributed Processing (ODP) describes systems that support heterogeneous distributed processing both within and between organizations through the use of a common interactive model. The International Standards Organization (ISO) and the International Telecommunication Union – Telecommunications Standardization Sector (ITU-T) have developed the Reference Model of Open Distributed Processing (RM-ODP) to provide a coordinating framework for the standardization of ODP by creating an architecture, which supports distributing, internetworking, interoperability, and portability. RM-ODP is a standardized framework for modeling such systems. [HERR97] The goal of RM-ODP is to achieve the following:

- Portability of application across heterogeneous systems
- Interlocking between ODP systems; that is, meaningful exchange of information and the convenient use of functionality throughout the distributed system
- Distribution transparency; that is, hide the consequences of distribution from both the applications programmer and user [ENTE99]

The following distribution transparencies defined in RM-ODP [ISO395]:

- Access transparency – hides the differences in data representation and procedure calling mechanism to enable interworking between heterogeneous computer systems
- Failure transparency – masks the failure and possible recovery of objects, to enhance fault tolerance
- Location transparency – masks the use of physical addresses, including the distinction between local and remote
- Migration transparency – masks the relocation of an object and its interfaces from other objects and interfaces bound to it
- Relocation transparency – hides the relocation of an object and its interfaces from other objects and interfaces bound to it
- Replication transparency – maintains consistency of a group of replicated objects with a common interface and is used to enhance performance and availability
- Persistence transparency – masks the deactivation and reactivation of an object
- Transaction transparency – hides the coordination required to satisfy the transactional properties of operations

The reference model provides the "big picture" that organizes the pieces of an ODP system into a coherent whole. It does not try to standardize the components of the system or unnecessarily influence the choice of technology.

The RM-ODP architecture uses the notion of viewpoints as an abstraction mechanism. The viewpoints on an ODP system and its environment are as follows:

- Enterprise viewpoint (purpose, scope, and policies)
- Information viewpoint (semantics of information and information processing)
- Computational viewpoint (functional decomposition)
- Engineering viewpoint (infrastructure required to support distribution)
- Technology viewpoint (choices of technology for implementation) [FEUS99]

Specifying an ODP system using the viewpoint languages allows for an otherwise large and complex specification of an ODP system to be separated into manageable pieces, each focused on the issues relevant to a member of the development team. For example, the systems analyst works with the information specification while the systems programmer is concerned with the engineering viewpoint. Figure 1 shows how the RM-ODP viewpoints can be related to the software engineering process of requirements analysis, functional specification, design, and implementation.

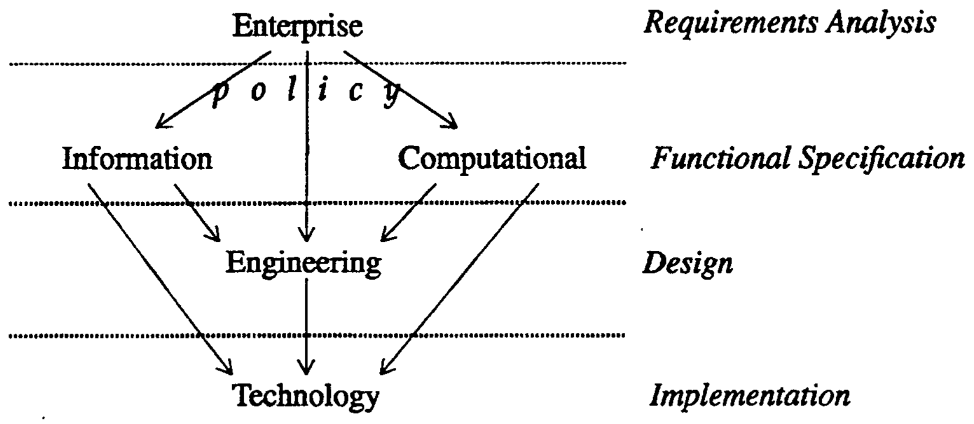

Figure 1. RM-ODP Viewpoints and Software Engineering

The five viewpoints are discussed in the following sections.

B. ENTERPRISE VIEWPOINT

The enterprise viewpoint focuses on the purpose, scope, and policies for the system. In the enterprise viewpoint, social and organizational policies can be defined in terms of objects, communities, and roles of the objects within the communities.

1. Objects

Objects are classified as either "active" or "passive." Some examples of active objects are bank managers, tellers, and customers. Examples of passive objects are bank accounts and money. [ISO395]

2. Communities

Communities are groupings of objects intended to achieve some purpose. An example of a community is a bank branch. The bank branch consists of objects (bank manager, tellers, and bank accounts), which provide banking services to a geographical area (purpose). [ISO395]

3. Roles of Objects

Roles of the objects within the communities are expressed in terms of policies. Policies dictate the behavior of objects within a community. Policies have varying scope. They may apply across a community, to a role, or to a single action type. Policies can be either implicit or explicit. Implicit policies are derived from the specification of a role, enterprise object, or action. An explicit policy is one for which policy is specified in its own right. [ISO395]

Policies consist of three components: 1) Permissions, 2) Prohibitions, and 3) Obligations. Permissions describe what can be done; for example, money can be deposited in an open bank account. Prohibitions describe what must not be done; for example, customers must not withdraw more money than they have in their bank account. Obligations describe what must be done; for example, the bank manager must advise customers when the interest rate changes. Permissions and prohibitions are defined by actions, roles involved in that action, predicates on behavior, and an authority, which either grants the permission or imposes the prohibition. [MICH99]

4. Enterprise Language

The enterprise language is a structured set of concepts, rules, and structures for the specification of an ODP system using the enterprise viewpoint. It is specifically concerned with performance actions that change policy, such as creating an obligation or revoking permission. In a bank, the changing of interest rates is a performance action since it creates obligations on the bank manager to inform customers of the change. However, obtaining an account balance is not a performance action because obligations, permissions, and prohibitions are not affected. [ISO195]

By imposing an enterprise specification of an ODP application, policies are determined by the organization rather than imposed on the organization by technology (implementation) choices. For example, a customer should not be limited to having only one bank account because it is more convenient for the programmer to implement.

C. INFORMATION VIEWPOINT

The information viewpoint focuses on the semantics of information and information processing. The information viewpoint is used to describe the information required by an ODP application through the use of schemas, which describe the state and structure of an object. An example of an information viewpoint is the information about a bank account object, which consists of a schema containing the balance and the "amount withdrawn today." [ISO395]

A *static schema* captures the state and structure of an object at some particular instance; for example, at midnight, the amount withdrawn today is $0. An *invariant schema* restricts the state and structure of an object at all times; for example, the amount withdrawn today is less than or equal to $500. A *dynamic schema* defines a permitted change in the state and structure of an object; for example, a withdrawal of \$*n* from an account decreases the balance by \$*n* and increases the amount withdrawn today by \$*n*. Dynamic schemas are constrained by any existing and applicable invariant schemas. [ISO395]

Schemas can also be used to describe relationships or associations between objects; for example, the static schema "owns account" could associate each account with a customer. A schema can be composed from other schemas to describe complex or composite objects; for example, a bank branch consists of a set of customers, a set of accounts, and the "owns account" relationships. The information specification of an ODP application can be expressed using a variety of methods; for example, entity-relationship models and conceptual schemas. [ISO395]

D. COMPUTATIONAL VIEWPOINT

The computational viewpoint focuses on the functional decomposition of the system into objects, which interact at interfaces. The computational viewpoint is used to specify the functionality of an ODP application in a distribution-transparent manner. RM-ODP's computational viewpoint is object-based. The objects encapsulate data and methods, offer interfaces for interaction with other objects, and offer multiple interfaces. [ISO395]

The computational viewpoint further defines the objects within an ODP system, the activities within those objects, and the interactions that occur among objects. Most objects in a computational viewpoint describe application functionality, and these objects are linked by bindings (e.g. a communication path) through which interfaces occur. Binding objects are used to describe complex interactions between objects. [ISO195] Figure 2 illustrates a bank branch object providing a bank teller interface and a bank manager interface. Both interfaces can be used to deposit and withdraw money, but accounts can be created only through the bank manager interface. Each of the bank branch object's interfaces is bound to a customer object.

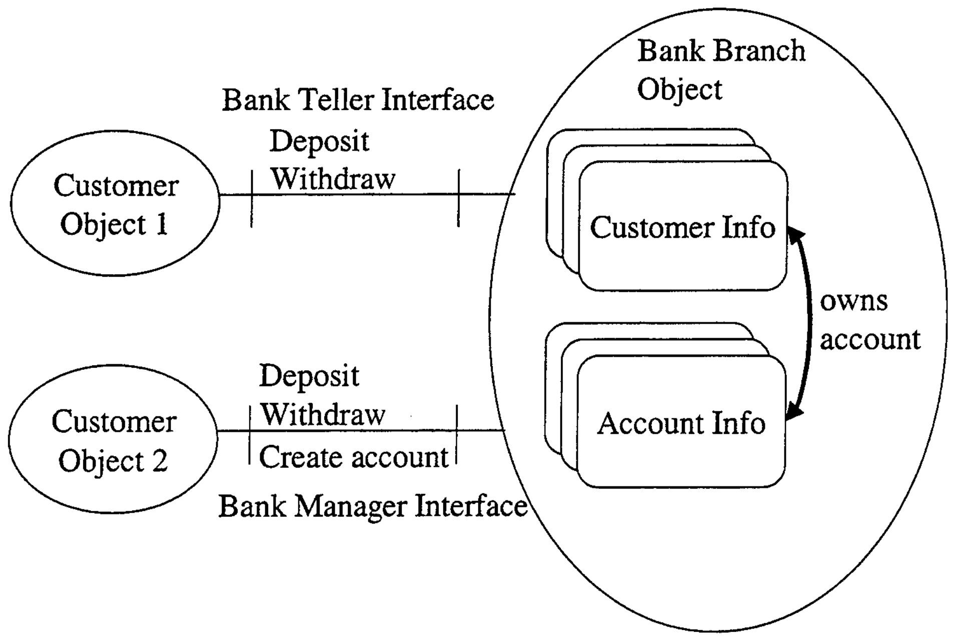

Figure 2. Bank Branch Object with Bank Manager and Bank Teller Interfaces

E. ENGINEERING VIEWPOINT

The engineering viewpoint focuses on the infrastructure required to support distributed interaction between objects in the system. The engineering viewpoint is not concerned with semantics of the ODP application, except to determine its requirements for distribution and distribution transparency. [ISO195] The computational viewpoint is concerned with when and why objects interact, while the engineering viewpoint is concerned with how they interact.

The fundamental entities described in the engineering viewpoint are objects and channels. Objects in the engineering viewpoint can be divided into two categories – basic engineering objects and infrastructure objects. Basic engineering objects correspond to the objects in the computational viewpoint. An example of an infrastructure object is a protocol object. A channel corresponds to a binding or binding object in the computational viewpoint. A channel provides the communication mechanism and contains or controls the distribution transparencies. [ISO395]

F. TECHNOLOGY VIEWPOINT

The technology viewpoint focuses on the choice of technology for implementation of the system. The technology viewpoint also describes the information required for testing. RM-ODP has very few rules applicable to technology specifications. [ISO395]

THIS PAGE INTENTIONALLY LEFT BLANK

III. RELATED WORK

This chapter describes some of the previous research most closely related to this thesis. The work of seven groups of authors is highlighted in the remainder of this section. A separate section for each group follows.

Finkelstein and Fuks propose a formal framework for understanding the software development process and consider in particular software development as cooperative work.

Nuseibeh, Kramer, and Finkelstein propose an object-based framework for the development of heterogeneous, composite systems. The framework uses viewpoints that represent agents having roles in and views of a problem domain. They also explore the use of inter-viewpoint rules to express the relationships between different viewpoints.

Finkelstein, Gabbay, Hunter, and Nuseibeh discuss the distributed development of specifications from multiple perspectives. The authors combine two areas of research: 1) the viewpoints framework for perspective development, interaction, and organization and 2) a logic-based approach to consistency handling.

Easterbrook and Nuseibeh discuss inconsistency management. The authors use viewpoints to identify consistency relationships, check for inconsistencies, and resolve conflicts; these are important steps in managing inconsistency in an evolving specification.

Kotonya and Sommerville discuss several viewpoint-oriented requirements approaches and give a description of an alternative, object-oriented, viewpoint-based approach.

Sibley, Michael, and Wexelblat discuss the use of a policy workbench to aid in analyzing proposed policy, maintain a policy database, and assist in policy enforcement. The authors discuss the requirements, preliminary design, and prototype implementation of such a workbench.

Lupu and Sloman focus on policy conflicts and the problems of conflict detection and resolution in distributed systems.

A. COOPERATIVE FRAMEWORK

Finkelstein and Fuks propose a formal framework for understanding the software development process and consider, in particular, software development as cooperative work. The requirements for a software development framework are reviewed and set in the context of an established development paradigm. The underlying model of cooperation, based on dialogue, is motivated and outlined. A formal scheme for expressing this model is introduced and the basic constructs are described. The authors also present a tool to investigate software development in this style. [FINK89]

The authors argue that a framework is fundamental to organizing the development of software as an engineering activity. The authors' discussion of a cooperative framework is divided into two parts. The first part outlines, from a software engineering perspective, the requirements and intuitions on which their approach is based. The second part presents and illustrates the basic elements of a formal model of software development. They try to closely integrate the formal approach to software development with software engineering concerns such as cooperation. [FINK89]

The primary requirement for a model of software development is that it be cooperative. Additionally, the model should not fail in the presence of conflict. The resolution of conflicts should be explicitly controlled and organized as part of the overall process of software development.

The authors' starting point of a formal model is the established paradigm of incremental refinement. The model is based on dialogue. The underlying vehicle for defining the framework is a systematic model of dialogue. The model has the following components:

1) View – a logical "perspective" in the dialogue in which a physical participant in a dialogue may have multiple views or "perspectives" that correspond with organizational roles
2) Commitment store – each view has a commitment store which holds its commitments within its domain; e.g. the rules
3) Board – the medium of communication between views which represents both a history of the dialogue and its current state
4) Database – holds "facts" internally or privately to the view, stored before the start of the dialogue, and contains knowledge, which does not have the full status of public engagement

5) Dialogue engine – interprets the general rules governing the dialogue in the context of the current dialogue and acts as the controller of the dialogue, on the part of the view [FINK89]

The authors analyzed the model with a case study of an automated travel ticket system. The authors believe that dialogue provides a basis for cooperative software development framework, especially in the area of defining requirements. [FINK89]

B. THE VIEWPOINTS APPROACH

Nuseibeh, Kramer, and Finkelstein discuss the problems of heterogeneity. They state that heterogeneity in large systems is inevitable and that no single development process or representation is sufficient for the development of large systems. Heterogeneity introduces problems of integration: 1) integration of the methods used to specify system requirements, 2) integration of the tools that support these methods, and 3) integration of the multiple specification fragments produced by these methods and tools. [NUSE93]

In the development of heterogeneous systems, the authors discuss a concept called the *multiple perspectives problem,* which they define as "the class of problems surrounding the development of composite systems by many development participants." [NUSE93] The authors use an object-based viewpoints framework to deal with the problems of integration. They use v*iewpoints* to describe the system development participants, their roles in the development process, and their views of the problem domain. [NUSE93] Viewpoints encapsulate development of systems into five different slots:

1) Style – notation or representation style used
2) Domain – describes the area of concern of the viewpoint with respect to the overall system under development
3) Work plan – development actions and strategy that use the notation defined in the style slot
4) Specification – describes the viewpoint domain in the notation described in the style slot and developed using the strategy described in the work plan slot
5) Work record – contains the development state and history of the specification produced in the specification slot and allows traceability

The authors define a *viewpoint template* as a viewpoint type in which only the style and work plan slots have been defined. When instantiated, a viewpoint template becomes a viewpoint, which can then be expanded to produce the specification for a particular domain. A viewpoint template is therefore a reusable description of a development technique, which can be instantiated many times to produce different viewpoints.

The style slot of each template may be represented in terms of objects and relations, with each having attributes with types and values. The work plan slot has four generic categories of development actions:

1) Assembly actions – the basic actions required to build (assemble) a specification in the current representation style

2) In-viewpoint check actions – actions that can be performed to check that a specification is syntactically consistent

3) Inter-viewpoint check actions – actions that can be performed to check the consistency between (overlapping and interacting) specifications residing in different viewpoints and may also be used to transfer information between viewpoint specifications

4) Viewpoint trigger actions – actions that must be performed in order to create new viewpoints (instantiated viewpoint templates)

The authors attempt to integrate multiple requirements specification viewpoints. In doing so, they state that overlaps must be identified and expressed, complementary participants must interact and cooperate, and contradictions must be resolved. The authors address *Inter-viewpoint* communication as a vehicle for viewpoint integration. The authors argue that "successful inter-viewpoint communication holds the key to achieving integration in a heterogeneous, possibly distributed, setting." There is a need to express relationships between viewpoints, enact these relationships (check consistency and transfer or transform information), and resolve conflicts (if and when it is necessary to do so). [NUSE93]

During development the different viewpoint specifications may be overlapping and/or inconsistent with each other. Inconsistency is tolerated in the viewpoints approach and not checked or corrected except as needed. Inter-viewpoint rules are then defined to check consistency and transfer and transform information between viewpoint specifications. The rules describe the relationships and specify when they may be invoked and how they should be applied. The rules are defined in the viewpoint template work plans and describe relationships between viewpoints that have not yet been created. The authors define the rules in great detail. The inter-viewpoint rules are invoked by the owner (development participant) of the viewpoint in which they reside. The authors discuss three approaches to rules invocation:

1) "The constrained" – in which rules are constantly invoked
2) "Pragmatic" – in which rule invocation may be turned on and off by the user
3) "Process-oriented" – in which the process model of the viewpoints guides rules invocation [NUSE94]

The authors state that "ideally, a method designer would be provided with a predefined library of relationships at his/her disposal, with the option of defining any further relationships if required." [NUSE94] There are two modes of application of an inter-viewpoint rule:

1) Check Mode – in which the relationship question is asked (does the relationship hold between the two viewpoints?). The relationship either holds or conflict resolution must be performed to make it hold.
2) Transfer Mode – in which the function is applied to transfer and transform information from one viewpoint to another so that the relation will hold between them. [NUSE94]

An invoked inter-viewpoint rule is normally applied in "check mode" to begin with, after which a "transfer mode" may be entered into if the rule failed. Information transfer is viewed as a form of conflict resolution.

The authors use a generic computer-based prototype environment called *The Viewer* to support the viewpoints framework. The Viewer provides tools for viewpoint construction and deployment. [NUSE92]

C. INCONSISTENCY HANDLING

Finkelstein, Gabbay, Hunter, and Nuseibeh discuss the distributed development of specifications from multiple perspectives. Using multiple perspectives leads to problems of identifying and handling inconsistencies between perspectives. The authors combine two areas of research: 1) the viewpoints framework for perspective development, interaction, and organization and 2) a logic-based approach to consistency handling. [FINK94]

The authors do not believe that it is possible to maintain absolute consistency between perspectives at all times. It is often undesirable to enforce consistency, “particularly when this constrains the specification unnecessarily or entails loss of design freedom by enforcing an early resolution.” [FINK94]

The authors provide an overview of inconsistency handling using the viewpoints framework. The authors also illustrate and discuss the identification and handling of inconsistency using simple library system example. They believe that inconsistency handling should be based on logic. They state that “the problem of inconsistency handling in the viewpoints framework can then be viewed as being equivalent to inconsistency handling in distributed logical databases.” [FINK94]

D. INCONSISTENCY MANAGEMENT WITH VIEWPOINTS

Easterbrook and Nuseibeh discuss inconsistency management. The authors state that "inconsistency is important in the software specification process, because, on the one hand, inconsistent specifications cannot be satisfied, whereas on the other, it is hard to achieve consistency in a large specification written by a team." [EAST96] They believe that proper management of inconsistencies lead to a more robust specification, because inconsistencies provide important clues about missing information. They show that viewpoints can be used as the cornerstone of effective inconsistency management. [EAST96]

They define *inconsistency* to be "any situation in which two parts of a specification do not obey some relationship that should hold between them." [EAST96] Inconsistency classification focuses on identifying the kind of inconsistency that has been detected in a specification. Inconsistencies are classified in two ways: 1) according to cause, the result of mistakes such as typographic error or conceptual disagreements and 2) into different predefined kinds of inconsistency that may arise in programming.

The authors define five ways of dealing with inconsistencies:

1) Ignore – appropriate if inconsistency is isolated and does not prevent further development, or the level of risk does not justify fixing it
2) Delay – used when further information is required, but is not immediately available
3) Circumvent – used to disable or modify the rule that was broken to get around the inconsistency

4) Ameliorate – appropriate when steps are needed to improve the situation but not necessarily remove the inconsistency, also called incremental resolution

5) Resolve – used to repair the inconsistency immediately [EAST96]

An inconsistency implies missing information. Inconsistency management provides a route to improved understanding of requirements. A key problem is tracking known inconsistencies and recording development information.

They base their work on a framework for distributed software engineering, in which multiple perspectives are maintained separately as distributable objects called v*iewpoints,* which are defined in [NUSE92], [NUSE93], and [NUSE94]. The framework is independent of any software development method.

In the authors' framework, there is no requirement for changes to one viewpoint to be consistent with other viewpoints. [EAST96] Inconsistencies are tolerated throughout the software development process. They focus on the management of inconsistencies. Consistency checking and resolution are still required, but they are delayed until an appropriate time. To manage inconsistencies, the relationships between viewpoints are clearly defined. Consistency checking is performed by applying a set of rules which express the relationships; these rules should hold between particular viewpoints.

The authors' use state transition diagrams to specify the required behavior of a device. The method permits partitioning of the state transition diagram describing a single device into separate viewpoints, such that the union of all of the viewpoints describes all of the states and transitions of the device. [EAST96]

The method provides the following:

1) The notation for expressing states and transitions diagrammatically
2) A partitioning step that allows a separate diagram to be created to represent a subset of the behaviors of a particular device
3) A set of consistency checking rules which test whether partitioned diagrams representing the same device are mutually consistent
4) An analysis step that allows two viewpoints to be treated as separate devices that interact
5) A further set of consistency checking rules which test whether interacting devices whose transitions have been linked together exhibit consistent behaviors

The authors also discuss *The Viewer*, a prototype computer-based environment and associated tools (discussed in Nuseibeh, Kramer, and Finkelstein's work), which has been constructed to support the viewpoints framework. *The Viewer* has two distinct modes of use: method design and method use. Method design involves creation of viewpoint templates; these are viewpoints for which only the representation style and work plan slots are defined. In method use, viewpoints are instantiated from the templates to represent the various perspectives. Each viewpoint is a self-contained specification development tool. The Consistency Check in *The Viewer* allows users to invoke and apply in- and inter-viewpoint consistency rules and record the results of all the checks in the work record of the affected viewpoint. [EAST96]

The authors state that viewpoints facilitate separation of concerns and the partitioning of software development knowledge. [EAST96] Partitioning is only useful if relationships and dependencies between partitions can be defined. The authors showed how relationships could be defined as part of a collection of viewpoints. They also demonstrated how inconsistencies identified by checking relationships could be resolved. According to the authors, identifying consistency relationships, checking inconsistency and resolving conflicts are all important steps in managing inconsistency in an evolving specification. [EAST96]

E. OBJECT-ORIENTED VIEWPOINT-BASED APPROACH

Kotonya and Sommerville discuss several viewpoint-oriented requirements approaches and give a description of an alternative object-oriented viewpoint-based approach. They use an automated teller machine (ATM) case study to analyze the various approaches.

The authors state that the "task of modeling a user's needs involves the capture, analysis and resolution of many ideas, perspectives and relationships at varying levels of detail." [KOTO92] The authors have developed a two-layered approach to system modeling. The first layer, the *viewpoint layer*, is concerned with "identifying relevant viewpoint entities in the system environment and associating them with the services they require." The second layer, the *system layer*, is concerned with "identifying system entities responsible for the provision of services to the viewpoint layer." In their research, the authors only discuss the viewpoint layer. [KOTO92]

The authors discuss and analyze six viewpoint-oriented approaches developed by other researchers, which include:

1) Structured Requirements Definition (SRD)
2) Structured Analysis and Design Technique (SADT)
3) Controlled Requirements Expression (CORE)
4) Viewpoints-Oriented Software Development (VOSD)
5) Viewpoint Analysis
6) Other Methods

1. Structured Requirements Definition (SRD)

SRD is based on "defining the application context." [KOTO92] It consists of a four-step process:

1) Define a user-level data-flow diagram by interviewing each individual in the organization to be analyzed who currently performs some useful task. Record the input/output as a data-flow model.
2) Combine all like user-level data-flow diagrams to create one integrated data-flow diagram. Conflicting data can be resolved at this stage.
3) Define the application-level data-flow diagram by drawing a dotted line around the part of the combined user data-flow diagram that corresponds to the organization being analyzed.
4) Define the application-level functions by showing the order that comprises each function being performed by the organization being modeled.

SRD viewpoints are individuals in the organization being specified that are manually performing a function which will be automated. SRD does not work well if the system being specified does not involve the automation of several manual tasks.

Kotonya and Sommerville list the shortcomings of the SRD approach. They state that SRD has an intuitive, rather than defined, notion of a viewpoint, making it the secondary rather than primary technique used in the methodology. The authors also state that "viewpoint analysis does not extend beyond data sourcing and sinking in SRD." [KOTO92]

2. Structured Analysis and Design Technique (SADT)

SADT is based on a data-flow model that views a system as a set of activities. "A rectangular box representing the system's most abstract activity, together with a set of data input, data output and control arrows, forms the starting point for functional decomposition." [KOTO92] Consistency checking involves "matching the set of inputs and outputs at each successive functional level with the higher function." [KOTO92]

The authors also discuss the shortcoming of SADT. Like SDR, the SADT viewpoint is not defined; it is an intuitive extension of the underlying modeling technique. SADT viewpoints are also not analyzed beyond being seen as data sources and sinks. SADT does not provide a framework for control information between viewpoints.

3. Controlled Requirements Expression (CORE)

CORE is a functional requirements definition method based on a viewpoint approach. It is one of the few requirements definition methods that explicitly uses a viewpoint approach. CORE defines viewpoints at two levels. The first level consists of all entities that interact or affect the system in some way. The second level is concerned with distinguishing between defining and bounding viewpoints. “Bounding viewpoints are entities that interact indirectly with the system, whereas defining viewpoints are sub-processes of the system, viewed in a top-down manner.” [KOTO92]

The CORE methodology consists of steps: 1) viewpoint identification, 2) viewpoint structuring, 3) tabular collection, 4) data structuring, 5) single viewpoint modeling, 6) combined viewpoint modeling, and 7) constraints analysis.

In step 1, the viewpoint identification step, functional and non-functional viewpoints are identified. Step 2, the viewpoint structuring step, provides a framework for requirements capture and analysis. It involves iterative decomposition of the system into a hierarchy of functional sub-systems, using a top-down decomposition. Step 3, the tabular collection step, is a mechanism for gathering information about a viewpoint. Each viewpoint is considered with respect to the following: 1) the action it performs, 2) the input data used for these actions, 3) the output data derived, 4) the sources of the data, and 5) the destinations of the data. Tabular collections are required to resolve omissions and conflicts in information across viewpoints. Steps 4 through 7 are not discussed in detail in the authors' work.

Kotonya and Sommerville state that the major weakness in CORE is its poorly defined notion of a viewpoint. Since the CORE viewpoint can be any entity that interacts with the system, it is difficult to say what is and is not a valid viewpoint.

4. Viewpoints-Oriented Software Development (VOSD)

In viewpoints-oriented software development (VOSD), Finkelstein proposed the use of viewpoints as a way of managing the software development process. VOSD uses viewpoints to "capture the role and responsibility performed by a participant at a particular stage of the software development process. Viewpoints are identified by the role of the participant, the domain relevant to their interest and the knowledge about the domain." [KOTO92] A viewpoint is a combination of a style or representation scheme in which the viewpoint expresses what it sees, the domain it sees, a specification, a work plan that defines the conditions under which the specification can be changed, and a work record. A template is defined as a viewpoint with only the style and work plan. [KOTO92] Viewpoints are discussed in greater detail in section B of this chapter.

VOSD is both a "requirements approach and an approach to facilitate distributed software development." [KOTO92] According to Kotonya and Sommerville, VOSD fails to provide a firm framework for resolving the conflicts between representations that have little correspondence. They also state that VOSD provides no obvious way of integrating functional and non-functional requirements and capturing the different classes of entities.

5. Viewpoint Analysis

Leite uses a viewpoints resolution approach as a means for early validation in the process of requirements elicitation. [KOTO92] The objective is to identify and classify problems related to correctness and completeness. Leite defines a *viewpoint* as "standing or mental position used by an individual when examining a universe of discourse (the overall context in which the software will be developed, including all the sources of information and the people involved)." [KOTO92] A *perspective* is defined as "a set of facts observed and modeled according to a particular aspect of reality." [KOTO92]

Viewpoint analysis uses three modeling aspects: the data perspective, the actor perspective, and the process perspective. A *view* is defined as an integration of these perspectives. View integration is achieved by a view-construction process. [KOTO92]

Leite uses a viewpoint language (VWPL) to represent the viewpoints. Kotonya and Sommerville state that Leite associates viewpoints with the roles of the people in analyzing the problem domain. They also state that VWPL is not a requirements language and that "its main objective is to register early results of the fact elicitation process in the requirements exercise." [KOTO92]

6. Other Methods

Kotonya and Sommerville also discuss two other approaches to requirements definition. Van Lamsweerde, Fickas, and Dardenne use a goal-directed approach to resolve conflicts and develop multiple viewpoints. Dubios, Hagelstein, and Rifuat developed an approach that focuses on the larger system formed by the computer and its environment, rather than on the required system software. [KOTO92]

Kotonya and Sommerville do not view Van Lamsweerde, Fickas, and Dardenne's approach or Dubios, Hagelstein, and Rifuat's approach as viewpoint oriented and, therefore, do not discuss them in detail.

7. Viewpoint-oriented Analysis (VOA)

Kotonya and Sommerville discuss their approach, which is an object-oriented viewpoint-based approach to requirements definition called viewpoint-oriented analysis (VOA). "VOA extends and enriches the traditional data-sink/source orientation of current viewpoint approaches to incorporate aspects of viewpoint classification and different modes of interaction." [KOTO92] The VOA framework supports:

1) Requirements capture and resolution
2) Classifying and analyzing viewpoint-system interaction
3) Integrating functional and non-functional aspects of requirements

VOA supports the identification of key system objects, their attributes, and their methods. The authors focus on the viewpoint layer. VOA includes four main stages: 1) viewpoint identification, 2) viewpoint structuring and decomposition, 3) information collection, and 4) reconciliation of information across viewpoints. [KOTO92]

VOA identifies a viewpoint as "an external entity that interacts with the system being analyzed, but one that can exist without the presence of the system." [KOTO92] Viewpoints fall into two classes:

1) Active viewpoints – entities that initiate some form of interaction between themselves and the system either by requesting services or providing control information to the system.
2) Passive viewpoints – essentially data stores or sinks. Any interaction between them and the system is initiated by the system.

Control information defines and describes how the environment controls the system and how the system controls the environment. In VOA, there are two levels of control, the viewpoint level and the system level. At the viewpoint level, controls are concerned with "the description of signals to start and stop specific services." [KOTO92] At the system level, controls are concerned with "associating control information described at the viewpoint level with entities in the system responsible for the provision of services." [KOTO92]

Viewpoint structuring provides the engineer with a tool for splitting the analytic tasks associated with viewpoints into a number of specification levels. A viewpoint is a hierarchy with differing levels of abstraction or specialization. The top level contains the most abstract description, and the lowest is most specialized. Each lower level inherits the services, attributes, control information, and non-functional constraints of the more abstract viewpoints. The sub-viewpoints may have their own constraints on inherited services and may also have their own particular service requirements.

VOA uses standard forms to collect information associated with a viewpoint. Information reconciliation is used "to ensure that all the required services are provided for, and that omissions or conflicting information can be identified and problems resolved." [KOTO92] According to Kotonya and Sommerville, VOA provides a powerful framework for integrating functional and non-functional aspects of requirements.

F. POLICY WORKBENCH

Sibley, Michael, and Wexelblat discuss the use of a policy workbench to aid in analyzing proposed policy, maintain a policy database, and assist in policy enforcement. The authors discuss the requirements, preliminary design, and prototype implementation of such a workbench. The objective of their research was to demonstrate computer support for a policy maker, enforcer, and user. The authors state that policy is often embedded in a system. [SIBL92]

The following are the requirements for a general policy workbench:

1) "The workbench outputs must be understandable by almost anybody with the domain knowledge in which the system is to be used." [SIBL92]
2) "The underpinnings of the policy representation must be based on appropriate formalisms but not be domain specific." [SIBL92]

The authors define five classes of people who need to deal with organizational policy: 1) policy maker, 2) policy maintainer, 3) policy implementer, 4) policy enforcer, and 5) policy user or designer of a system.

One of the requirements for a policy workbench is "the ability to represent policy in a way that can be supported by formal analysis." [SIBL92] They discuss possible problems in policy representation:

1) Tend to be abstract and stated in imprecise language
2) Are often incomplete and inconsistent
3) Are often interdependent and nested
4) May be very large

5) Need to represent time as an important component

6) Require a statement of intent to be fully representable

7) Fall along a wide spectrum of scope, from general to specific [SIBL92]

The policy workbench is comprised of the following three tools [SIBL92]:

1) "A *theorem and assertion analyzer* [entering and exercising policy] to check inputs, stated as axioms and theorems"

2) "A *rule compiler-generator-interpreter* [selecting, merging and generating parts of systems] to produce an executable component of the system"

3) "An interactive *policy structurer and selector* [aiding in understanding and applying policy] to check what rules are applicable to a given situation and preprocessing the rules into pre- and post-conditions"

The authors placed emphasis on "developing and experimenting with a prototype policy workbench that supports each software development phase." [SIBL92]

G. CONFLICTS IN POLICY-BASED DISTRIBUTED SYSTEMS

Lupu and Sloman focus on policy conflicts and the problems of conflict detection and resolution. Distributed systems contain a large number of objects. New components and services are added and removed from the system dynamically, which changes the requirements of the management system over its lifetime. [LUPU97]

The authors state that policies are a means of specifying and influencing management behavior within a distributed system, without coding the behavior into the management agents. The authors' approach is aimed at specifying implementable policies. They define two types of policies:

1) Authorization policies – security policies related to access control and specify what activities a subject is permitted or forbidden to do to/with a set of target objects
2) Obligation policies – specify what activities a subject must or must not do to a set of target objects and define the duties of the policy subject

The authors permit specification of both positive and negative authorization policies and require explicit authorization. They state that conflicts can arise in the set of policies due to omissions, errors, or conflicting requirements of the administrators specifying the policies. For example, an obligation policy may define an activity a manager must perform, but there is no authorization policy to permit the manager to perform the activity. Conflicts can also arise between positive and negative policies applying to the same objects. Whenever multiple policies apply to an object, there is the potential for some form of conflict, but multiple policies are essential. Many policies specified for the management of a large system specify exceptions to more general policies. The system may have to resolve conflicts such as exceptions to normal authorization policies. A precedence relationship is established between policies to resolve conflicts.

The authors use roles as the mean of grouping policies related to a particular manager position. Then managers can be assigned or removed from the position without changing the policies. They also define the relationships between roles with regard to the use of shared resources or with regard to organizational structure. The authors use roles and inter-role relationships to provide a scope for the conflict detection and help to limit the number of policies that must be examined in order to detect conflicts. They focus on techniques and tool support for off-line (static) detection and resolution, although some conflicts can be detected only at run-time (dynamic). [LUPU97]

The authors state that policies are interpreted by automated manager agents, so the behavior of the agents can be modified dynamically by changing policy rather than re-coding. Obligation policies can be either used in conjunction with authorizations in order to ensure the integrity of the system or to specify the actions a component must initiate in response to changes in its internal state or environment. [LUPU97]

The authors discuss the details of the domains, policies, and roles, which form their management framework. They also discuss the types of inconsistencies and the policy conflicts that they need to detect. Next they explain their approach to conflict detection and conflict resolution based on policy-precedence relationships. [LUPU97]

The authors state that *domains* are used to partition objects in a large system according to geographical boundaries, object type, management functionality, responsibility, and authority. Domains may also be used to group objects in order to apply a common policy to a set of objects. A *sub-domain* is a domain which is a member of another domain or parent domain. [LUPU97]

The authors discuss in detail the notation used to specify policies. Authorization (A) and Obligation (O) policies may be positive or negative. The authors also provide some example policies and show how conflicts can occur due to positive and negative policies. [LUPU97]

They define *modality conflicts* as inconsistencies in the policy specification which can occur when two or more policies with modalities of opposite sign refer to the same subjects, actions, and targets. There are three types of modality conflicts:

1) O+/O-: the subjects are both required to and required not to perform the same actions on the target objects
2) A+/A-: the subjects are both authorized and forbidden to perform the actions on the target objects
3) O+/A-: the subjects are required but forbidden to perform the actions on the target objects; obligation does not imply authorization

The authors focus their attention on a static conflict detection tool, which assists the users in specifying policies, roles, and relationships. They discuss some principles for the detection of modality conflicts and present an implementation of the conflict detection tool.

They propose that using a policy precedence relationship can reduce the number of conflicts between policies and permit inconsistent specifications. The following are the principles for establishing precedence:

1) Negative policies always have priority
2) Assigning explicit priorities

3) Distance between a policy and the managed objects – priority is given to the policy applying to the closer class in the inheritance hierarchy when evaluating access to an object reference in a query

4) Specificity related to domain nesting – can be used within conflict detection to reduce the number of potential conflicts

The authors state that modality conflicts can be detected purely by syntactic analysis of the policies. Application-specific conflicts arise from the semantics of the policy and are specified in terms of constraints on attribute values of permitted policies; they call these meta-policies. The authors use a prototype conflict detection tool to detect overlaps between policies and optionally apply domain nesting-based precedence. [LUPU97]

The authors found that assigning positive or negative policies were limiting and not as intuitive as the precedence-based approach on specificity. They actually only implement positive authorizations and remove negative authorizations by refinement of the policies. They assume all actions are prohibited unless explicitly authorized. Their policies are not global; the policies are interpreted by explicitly specified subjects (for obligations) and targets (for authorization). [LUPU97]

The authors discuss Sibley, Michael, and Wexelblat's Policy Workbench. They state that the Policy Workbench uses automated tools to specify and analyze policies. The authors state that this approach is complex due to the generality of their policies. The Policy Workbench is not based on precedence relationships.

In summary, Lupu and Sloman presented the integration of a conflict detection tool in a role and policy-based framework. They performed off-line, static analysis of a set of policies to determine two types of conflicts:

1) Modality conflicts – arising from positive and negative policies, which can be checked by analyzing the syntax of the policies
2) Application specific conflicts – need to be expressed by external constraints or meta-policies

They make use of precedence relationship based on the specificity of the policies with respect to domain nesting to reduce the number of potential overlaps indicated to a user and allow inconsistencies between policies to exist within a system. The authors implemented a prototype role framework which supports distributed policy and domain servers and analysis of a set of policies. Their approach is "to detect as many conflicts as possible at time of specification, rather than leaving them to be detected at run-time." [LUPU97] The user can then modify the policies to remove conflicts. The authors implemented their approach using a CORBA-based distributed programming environment.

The authors concentrated on the static analysis of policies. They stated that an area needing more research is dynamic run-time conflict detection. Some constraints can only be evaluated at run-time, since they depend on object states or current time. [LUPU97]

IV. OVERVIEW OF THE BALLISTIC MISSILE DEFENSE SYSTEM

A. MOTIVATION

Ballistic missiles have been a threat to the U.S. and its military operations for over fifty years. The global proliferation of ballistic missile technology and weapons of mass destruction is one of the most immediate and dangerous threats to U.S. national security in the post-Cold War era. The proliferation of short-range ballistic missiles in the world today poses a direct, immediate threat to many of our allies and to some U.S. forces deployed abroad in defense of our national interests. Over time, the proliferation of longer-range missiles could pose a greater threat to the U.S. itself. [MOSH97]

The U.S. has already witnessed the willingness of countries to use theater-class ballistic missiles for military purposes; e.g., the use of Scuds during the Gulf War. Since 1980, ballistic missiles have been used in six regional conflicts. Strategic ballistic missiles, including intercontinental and submarine launched ballistic missiles (ICBMs and SLBMs), exist in abundance in the world today. There is also great concern from the emergence of a Third World long-range missile threat to the United States. [BMDOLINK]

Ballistic Missile Defense (BMD) plays a central role in U.S. national security strategy by supporting its defense and counterproliferation objectives. [BMDOLINK] The requirement for BMD stems from a strategy that requires the U.S. to maintain an overseas presence and the capability to respond to major regional conflicts despite the increasing danger posed by the proliferation of ballistic missiles.

> In a world of regional threats to the U.S., BMD affords the U.S. greater freedom of action to protect its interests and uphold its security commitments without fear of coercion. BMD can bolster the solidarity of coalitions and alliances, as it did during Desert Storm in 1991, and provide a response to crises without having to resort to offensive measures. Finally, BMD can strengthen the credibility of our deterrent forces and provide an essential hedge against the failure of deterrence. [BMDOLINK]

B. DESCRIPTION OF BMD SYSTEM

The Ballistic Missile Defense Organization (BMDO), an agency of the Department of Defense (DoD), is "responsible for managing directing, and executing the BMD Program." [BMDOLINK] The BMD program is a two-part missile defense program. First priority is given to Theater Missile Defense (TMD) to meet the existing threat to deployed U.S. and allied forces. Second priority is assigned to National Missile Defense (NMD) as a hedge against the emergence of long-range ballistic missile threats. [BMDOLINK]

1. Theater Missile Defense (TMD)

Theater Missile Defenses are designed to protect U.S. deployed troops and coalition forces. TMD systems must be able to deploy rapidly and be highly mobile. "Since the TMD threat is diverse with respect to range and capability, no single system can perform the entire TMD mission." [BMDOLINK] To adequately perform TMD, an integrated Family of Systems (FoS) is required. (See Table 1) "The FoS concept is an evolving, flexible configuration of interoperable weapon systems, Command and Control

(C2) centers, and internal/external sensors." [BMDO99G] Development of these systems is dispersed throughout the Services and Agencies. BMDO states that the FoS approach will ensure a defense in depth, utilizing lower-tier (low-altitude), upper-tier (high-altitude), and boost-phase defenses. [BMDO99G]

System Name	Deployment mode	Approximate radius of defended area	Intended to defend against:	Type of warhead
Lower-tier theater missile defenses:				
Patriot PAC-2 (upgraded)	Truck-mounted	10-15 kilometers	Short-range missiles with range up to 600 kilometers; Aircraft	Blast fragment
Patriot PAC-3	Truck-mounted	40-50 kilometers	Short-range missiles with range up to 1,500 kilometers; Aircraft	Hit-to-kill
MEADS (Medium-range Extended Air Defense System)	Truck-mounted	Less than 10 kilometers	Short-range missiles with range up to 600 kilometers; Aircraft	Hit-to-kill
Navy Area Defense	Ship-based	50-100 kilometers	Short-range missiles with range up to 600 kilometers; Aircraft	Blast fragment
Upper-tier theater missile defenses:				
THAAD (Theater High Altitude Area Defense)	Ground-based; transportable by aircraft	A few hundred kilometers	Short and medium-range missiles with range up to 3,500 kilometers	Hit-to-kill
Navy Theater Wide	Ship-based	More than a few hundred kilometers	Short and medium-range missiles with range up to 3,500 kilometers	Hit-to-kill
Boost-phase theater missile defenses:				
Airborne Laser	Airplane	Possibly huge	Short and medium-range missiles with range up to 3,500 kilometers	Directed energy

Table 1. US Theater Missile Defenses
This table lists the main U.S. theater missile defense systems that are either operational (Patriot PAC-2) or under development. [MOSH97]

a) *Lower-Tier Defenses*

Lower-tier defenses are designed to intercept missiles low in the atmosphere (at altitudes less than approximately 20 kilometers). [MOSH97] The interceptors must intercept their targets in the atmosphere because the interceptors maneuver to their target by using fins to steer through the air. Lower-tier defenses have relatively slow-flying interceptors that cannot fly very far before intercepting their targets; therefore, lower-tier defenses can cover only relatively small areas. Lower-tier defenses are designed to intercept short-range ballistic missiles, with ranges of up to roughly 600 to 1,500 kilometers, depending on the system. [MOSH97] In addition, these defenses are designed to shoot down aircraft and cruise missiles. [BMDO99G]

The United States currently has one lower-tier theater defense in operation, the Patriot PAC-2. The lower defense systems currently under development by the U.S. are the Patriot PAC-3, the Navy Area Defense, and the Medium-range Extended Air Defense System (MEADS). [BMDO99A]

(1) Patriot Advanced Capability-2 (PAC-2). According to Mosher, the PAC-2 is a transportable, truck-mounted system designed to defend small areas against aircraft and ballistic missiles with ranges of up to about 600 kilometers. [MOSH97] The interceptor uses a "blast fragmentation" warhead that is designed to explode once it gets within several meters of its target. During the Gulf War, the Patriot air defense system made its "famous battlefield debut against tactical ballistic missiles and helped defend coalition forces and Israeli territory against Iraqi Scud missile attacks." [BMDO99D] The current PAC-2 version is an upgraded version of the Patriot PAC-2 defense that was used during the Gulf War. Since the Gulf War, BMDO and the Army have significantly increased the effectiveness of the Patriot system by deploying the PAC-2 Guidance Enhanced Missile (GEM) which was developed to improve the Patriot's accuracy against short-range ballistic missiles. [BMDO99D]

(2) Patriot Advanced Capability-3 (PAC-3). Mosher describes the PAC-3 as a transportable, truck-mounted system designed to defend small

areas against ballistic missiles with ranges up to about 1,500 kilometers. [MOSH97] Unlike the PAC-2, this system does not use an explosive warhead. The PAC-3 uses a "hit-to-kill" interceptor (based on the earlier Erint missile), which is designed to destroy its target by hitting it directly. [MOSH97]

According to BMDO, the PAC-3 is a core TMD system with one of the highest priorities in the development of BMD systems. [BMDO99D] Its mission is to provide the lower tier of the BMD architecture. This includes defending troops and fixed assets from short and medium-range ballistic missiles, cruise missiles, and other air breathing threats such as fixed or rotary wing aircraft. To accomplish this mission, the PAC-3 system will be designed to be a "highly advanced missile defense system that can destroy enemy threats with hit-to-kill accuracy in the terminal phase of the threat missile's flight." [BMDO99D] "The PAC-3 system is planned to be interoperable with other Army and Joint systems, to provide a seamless missile defense in depth, and to be air-transportable to support rapid deployments." [BMDO99D]

PAC-3 systems contain four basic components: a radar set, an engagement control station (ECS), a launching station, and interceptors. The radar station provides warning and tracking of incoming threats and a continuous update link with in-flight interceptors. "The ECS computes fire solutions for the interceptor and provides fire control and a communications link with other Patriot units." [BMDO99D] The launch station transports, protects, and launches the missiles. Each launch station can be equipped with four GEM or earlier missiles. The PAC-3 missile uses its high maneuverability and hit-to-kill accuracy to destroy its target. [BMDO99D]

(3) Navy Area Defense. Navy Area Defense is a ship-based system designed to defend small areas against aircraft and ballistic missiles with ranges up to 600 kilometers. [MOSH97] Like the Patriot PAC-2, this system will use an explosive warhead. AEGIS cruisers and destroyers, equipped with a modified AEGIS combat system (ACS), will detect and track short-to-medium range TBMs and engage them with the Standard Missile-2 (SM-2) Block IVA interceptor. [BMDO99B]

The Navy Area Defense system "represents a lower-tier TMD capability that can take advantage of the strength and presence of the naval forces, and build upon the existing AEGIS infrastructure." [BMDOLINK] Naval vessels, which are routinely deployed worldwide, are currently in potential threat areas or can be rapidly redirected or repositioned. A Naval TMD capability can therefore be placed within a region of conflict to provide TMD protection for nearby land-based assets before hostilities erupt or before land-based defenses can be transported into the theater. [BMDO99B]

(4) Medium-range Extended Air Defense System (MEADS). The last of the lower-tier systems is the Medium-range Extended Air Defense System, formerly the Corps SAM [surface-to-air missile] program. MEADS is the only theater missile defense system under consideration to provide maneuver forces with 360-degree defense protection against the real and growing threat of short-range tactical ballistic missiles, cruise missiles and unmanned aerial vehicles. [MOSH97] This system

is intended to provide fundamental enhancements in tactical mobility, strategic deployability and operational capability. [BMDO99F]

MEADS is a truck-mounted system designed to be more mobile than the Patriot systems and to be deployed with ground troops as they move in the field. [MOSH97] By contrast, Patriot is designed to be operated from a single location for days at a time or longer. MEADS uses a hit-to-kill warhead and is designed to intercept ballistic missiles with ranges of up to 600 kilometers. MEADS is an international program under joint development with the United States, Germany, and Italy. [BMDO99F]

b) Upper-Tier Defenses

Upper-tier defenses are designed to intercept missiles high in the atmosphere or above the atmosphere. [MOSH97] This permits large ground areas to be covered. At the same time, upper-tier defenses are designed to intercept longer-range theater missiles, with ranges of up to 3,500 kilometers. Upper-tier defenses are also intended to be used as the first layer of a layered defense against short-range missiles, with the lower-tier theater defenses providing the second layer of defense. [BMDO99G]

The United States has two upper-tier defenses under development, Theater High-Altitude Area Defense (THAAD) and Navy Theater Wide (NTW). Both use hit-to-kill interceptors that maneuver to their target by using small thrusters to change their trajectory. [MOSH97] These interceptors operate at high altitudes where there is not enough air to enable them to maneuver by using fins. The interceptors currently under development use infrared sensors to detect the target and home on it. These sensors, which detect heat, will be blinded if they are used at low altitudes where the air resistance causes heating of the fast-flying sensor. Thus, early upper-tier interceptors will have a minimum intercept altitude below which they cannot intercept a target. [MOSH97]

(1) Theater High Altitude Area Defense System (THAAD). THAAD is a land-based system intended to defend large areas against missiles with ranges of up to 3,500 kilometers. [MOSH97] THAAD is designed to be transportable by aircraft and to intercept missiles high in the atmosphere (at altitudes above about 40 kilometers) or above the atmosphere. The THAAD interceptor has a top speed of about 2.5 kilometers per second. [BMDO99E]

The THAAD system is a critical element of the TMD FoS. [BMDO99E] THAAD is used to engage short, medium, and long-range ballistic missiles. The THAAD system's ability to intercept missiles at long range and high altitude (endo- and exo-atmospheric) will give U.S. forces the best chance to shoot down incoming missiles far enough out so that post-intercept debris will not harm U.S. troops, which is a vital concern if a missile carries a weapon of mass destruction. [MOSH97] This ability will also give the U.S. TBMD forces the time to judge the success of an intercept attempt and, if necessary, launch more interceptors from THAAD or other missile defense systems. As the upper tier of a two-tiered TMD architecture, THAAD provides near leak proof protection when employed with PAC-3 or Navy Area Defense. [MOSH97] THAAD system development started in 1992, and it is the most mature upper-tier system. It is expected to be fielded in 2007. Currently THAAD is undergoing testing and development. [BMDO99E]

The THAAD system consists of four principle components: truck-mounted launchers, interceptors, the THAAD Radar system, and the THAAD BM/C4I system. The purpose of the mobile launcher is to protect, transport, and fire the interceptors. "The interceptors will consist of a single stage booster and a kinetic kill-vehicle that will destroy targets by colliding with them." [BMDO99E] The THAAD radar will support surveillance, target tracking, and fire control functions, and provide a communications link with THAAD interceptors in-flight. THAAD's BM/C4I system will manage and integrate all THAAD components. [BMDO99E] The BM/C4I system will link the THAAD system to other missile defense and air defense systems to support a multi-tiered, highly effective, interoperable TMD architecture. The THAAD system will be transported by air on a C-141 cargo aircraft. [BMDO99E]

(2) Navy Theater Wide (NTW) Defense. NTW is a ship-based system intended to defend large areas against missiles with ranges of up to 3,500 kilometers. [MOSH97] NTW is designed to intercept medium- and long-range TBMs only above the atmosphere; it will use the LEAP (lightweight exo-atmospheric projectile) kinetic kill vehicle, which cannot intercept at altitudes below about 80-100 kilometers. [MOSH97] The system will not be able to intercept short-range missiles with a range less than about 300-400 kilometers, since they never reach an altitude of 80-100 kilometers. Navy Theater Wide is intended to intercept targets in the middle of their

trajectory (in mid-course) or, if the ship can position itself near the missile launch site, in the beginning of their trajectory shortly after the missile engine finishes burning (in ascent/boost phase). The interceptor has a speed of about 4.5 kilometers per second. [BMDO99C]

The system will be deployed on AEGIS ships, which use the SPY radar system. [BMDO99C] The NTW will be able to take advantage of the mobility of Navy AEGIS equipped ships and provide BMD protection to U.S. and coalition forces throughout the world. This is especially important in the early stages of conflict when land-based BMD assets are either unavailable or limited in number or location. [BMDO99C]

A second-generation system, Navy Theater Wide Block II, is also planned for deployment after 2010. This system will use a faster interceptor and a more powerful radar. [BMDO99C]

c) Boost-Phase Defenses

The United States is also developing systems to intercept missiles during the early, powered part of flight when the rocket booster is burning. This part of the missile's trajectory is called the "boost phase," and the ballistic missile defenses are termed boost-phase defenses. The advantage of boost-phase defenses is that they are designed to destroy the missile before the warhead and any decoys are released, so there would be only one target to destroy rather than potentially dozens or hundreds. [MOSH97] A boost-phase defense would also be able to destroy submunitions before they were released. Deploying chemical or biological weapons on numerous submunitions would be the best way for an attacker to distribute the agents and would simply overwhelm any mid-course or terminal defense system. [MOSH97] The disadvantage of boost-phase defenses is that the boost phase lasts for only a few minutes, and the interceptor must be able to make its intercept close to the launch site. There is currently one boost-phase theater defense under development, the Airborne Laser (ABL) [MOSH97]

(1) Airborne Laser (ABL). ABL is designed to attack short and medium-range missiles during their boost phase with a laser based on a modified Boeing 747 airplane. [MOSH97] The laser would be targeted on the missile, and if it shined on the same spot for long enough, it would weaken the metal surface by heating it to its structural-failure temperature, where the strength of the metal falls dramatically. For theater missiles, the airplane must be within several hundred kilometers of the missile the laser is attacking. It is generally assumed that the airplane would need to fly outside the borders of a country to avoid being shot down by air defenses; thus, this system will be incapable of attacking missiles launched from the interior regions of relatively large countries. [MOSH97] The exception would be in a conflict in which the United States had already established air superiority. In principle, the airborne laser would also be capable of causing a long-range missile to fail. Russian and Chinese land-based missiles are not assumed to be vulnerable to the ABL though, since they are based

far inland. However, the airborne laser could, in principle, threaten Russia's long-range SLBMs. [MOSH97]

2. National Missile Defense (NMD)

The NMD system is being developed to protect the United States from attack by a limited number of ICBMs armed with conventional, nuclear, biological, or chemical warheads. [MOSH97] Such limited attacks, ranging from a few to a few tens of missiles, fall into three categories:

1) A small accidental or unauthorized launch from Russia (or other former Soviet Union country)
2) A deliberate or unauthorized attack from China, which has some two dozen ICBMs with a range capable of reaching the United States
3) A deliberate attack from a hostile emerging missile state that might acquire long-range ballistic missiles [MOSH97]

The third threat, focused on North Korea, Iran, and Iraq, has emerged as the primary argument for a near-term NMD deployment. The NMD program has anticipated for some time the possibility that a rogue state could acquire ICBMs that could threaten the United States. This possibility was underscored by the August 1998 North Korean attempt to launch a satellite on a Taepo Dong-1 (TD-1) missile. [MOSH97] The launch demonstrated some important aspects of ICBM development, most notably multiple–stage separation. While the intelligence community expected a TD-1 launch for some time, it did not anticipate that the missile would have a third stage or that it would be used to attempt to place a satellite in orbit. [MOSH97] A three-stage variant of the TD-1, if successfully developed and deployed, could pose a threat to portions of the United States as well as to the territory of U.S. allies and friends. [BMDO99B]

The intelligence community's current view is that North Korea is more likely to develop the Taepo Dong–2 (TD-2) missile as a weapon. [BMDO00A] The TD–2 is a derivative of TD–1 technology, employing a larger first stage and the No Dong theater ballistic missile as the second stage. A two-stage TD-2 will have the range to reach Alaska, while a three-stage variant could bring most of the lower 48 states within range of North Korean ballistic missiles. The intelligence community believes North Korea could test a TD-2 at any time, unless it is further delayed for political reasons. Other rogue nations, particularly Iran, could test an indigenously developed ICBM in the latter half of this decade, using foreign assistance. These nations may also pursue a TD-type ICBM, possibly with North Korean assistance or purchase such a North Korean system outright, in the next few years. [MOSH97]

The NMD system being developed would defend all fifty states against a small number of ICBMs launched by a rogue state, such as North Korea. [BMDO00A] The NMD program is currently in its development phase. A decision whether to begin deployment of a limited national missile defense (NMD) system will be made in mid-2000 and will be "based on technology development, affordability, the potential threat, international treaty considerations, and competing defense priorities." [BMDOLINK]

The NMD system is an integrated collection of subsystems, referred to as "elements," that perform dedicated functions during an ICBM engagement. The system will include Ground-Based Interceptors (GBI), four types of long-range sensors (the

Defense Support Program satellites, Space-Based Infrared System satellites, an Upgraded Early Warning Radar (UEWR), and a Ground-Based X-Band Radar (XBR)), and a BM/C4I element. [BMDO00A]

The system will use GBIs topped with an exo-atmospheric kill vehicle (EKV) that is designed to destroy the incoming warhead by colliding with it at high speed. [BMDO00A] The GBI is a silo-based, three-stage, ICBM-class missile that delivers a separating EKV to an "acquisition point" above the atmosphere en route to engage a threat target. [BMDO00A] At this point, in a manner similar to upper-tier theater missile defense systems, the EKV uses an infrared seeker to acquire and track the target, firing divert thrusters (for terminal guidance and control) to achieve a direct hit on the targeted reentry vehicle. This collision will take place above the atmosphere, when the warhead is in the mid-course of its trajectory. [MOSH97]

The launch of an attacking missile would first be detected by U.S. early warning satellites. The existing satellites, known as Defense Support Program (DSP) satellites, use infrared sensors to detect the hot plume of a missile booster in the early stage of its flight. Beginning in 2004, the DSP satellites will be replaced by a new system of early warning satellites known as SBIRS-high (Space-Based Infrared System-high-earth orbit), which will also use infrared sensors to detect missile plumes but have improved capabilities. [BMDO00A] The data from the early warning satellites will be fed to the NMD Battle Management Center, to be located at Cheyenne Mountain in Colorado. [BMDO00A]

Once the booster finishes burning, the NMD system will use different sensors to detect the missile and any objects it releases, to track these objects accurately enough to guide the interceptors, and to discriminate the real warhead from decoys or other false targets. [MOSH97] These sensors include five existing early-warning radars, in Massachusetts, California, central Alaska, Greenland, and Britain, which will be upgraded to give them the ability to track targets accurately enough to guide interceptors. [BMDO00A] New XBRs designed specifically for NMD and with much greater discrimination capabilities will also be deployed. These ground-based radars will be supplemented by a space-based system of roughly 24 SBIRS-low (Space-Based Infrared System-low-earth orbit) missile-tracking satellites that are designed to provide track data accurate enough to guide interceptors without assistance from other sensors. [BMDO00A]

At some point in this process, the system must discriminate the actual warhead from the other objects. Otherwise, the NMD system, having a limited number of interceptors, would risk running out of interceptors if it attempted to fire at all the objects. [LEWI97] Because the NMD interceptors are designed to intercept their targets above the atmosphere, where there is no air resistance and where lightweight objects travel on the same trajectory as a heavy warhead, the system will be particularly vulnerable to countermeasures that use numerous lightweight decoys. [LEWI97]

The NMD Battle Management Center will integrate the information from the various sensors and decide which objects the system should try to intercept. The NMD system will then launch interceptors and guide them towards their targets. An In-Flight

Interceptor Communications Systems (IFICS), which will consist of several ground stations deployed at forward locations, will relay communications from the Battle Management Center to interceptors that have flown over the horizon. [MOSH97] As each interceptor nears its assigned target, it will release the EKV, which will use infrared and visible light sensors to detect the target and attempt to discriminate it from decoys or other false targets. [MOSH97] Finally, the EKV will home on the target and use thruster rockets to steer itself into the target.

To increase its odds of success, the NMD system would likely fire several interceptors at each target. To conserve interceptors, if time permits, the defense will use a shoot-look-shoot strategy, in which one or more interceptors are fired at the target, and after observing the results of the intercept attempts, additional interceptors are fired if necessary. Current plans reportedly call for firing four interceptors at each target. [BMDO00A]

The United States plans to build the NMD in three stages, with the capability of the system designed to increase in each stage. [BMDO00A] The first system configuration, called the Capability-1 (C-1) system, is designed to defend against an attack of a "few, simple" warheads. This initial system would be augmented to provide a Capability-2 (C-2) system, designed to defend against a "few, complex" warheads. The stated goal of the NMD program is to deploy a Capability-3 (C-3) system, designed to defend against "many, complex" warheads. The term "few" refers to five or fewer warheads. The dividing line between the terms "simple" and "complex" refer to the extent to which the attacker has incorporated countermeasures to fool or overwhelm the defense. [BMDO00A]

These three system configurations will differ from each other in several ways. The most obvious difference is in the number of interceptors. The C-1 system will have 20 interceptors deployed at a single site in the United States, the C-2 system will increase the number of interceptors at this site to 100, and the C-3 system will deploy a total of more than 100 interceptors at multiple sites in the United States. [BMDO00A] The number and types of sensors available to the NMD system in each case will also differ. The number of X-band radars will increase as the system evolved from the C-1 to the C-3 configuration, and the SBIRS-low sensors would first be deployed with the C-2 system. [BMDO00A]

The initial site will be either Grand Forks, North Dakota or central Alaska. The site not chosen for initial deployment will likely be used as a second site for the C-3 system. The Clinton administration has indicated it is leaning strongly towards an initial deployment in Alaska, as depicted in Table 2. [BMDOLINK]

	C-1 Configuration	C-2 Configuration	C-3 Configuration
Number of Interceptors Deployed in Alaska	20	100	125
Number of Interceptors Deployed in North Dakota	0	0	125
Upgraded Early-Warning Radars	Beale (Marysville, CA) Clear (Alaska) Cape Cod (Mass.) Fylingdales (England) Thule (Greenland)	Beale Clear Cape Cod Fylingdales Thule	Beale Clear Cape Cod Fylingdales Thule South Korea
X-band Radars	Shemya (Alaska)	Shemya Clear (Alaska) Fylingdales (England) Thule (Greenland)	Shemya Clear Fylingdales Thule Beale Cape Cod Grand Forks (ND) Hawaii South Korea
In-Flight Interceptor Communications Systems	Central Alaska Caribou (Maine) Shemya (Alaska)	Central Alaska Caribou Shemya Munising (Michigan)	Central Alaska Caribou Shemya Munising Hawaii
DSP or SBIRS-high?	Yes	Yes	Yes
SBIRS-low?	No	Yes	Yes
Source: US Ballistic Missile Defense Organization, March 3, 1999 viewgraph			

Table 2. Preliminary Architecture for C-1, C-2, and C-3 NMD Systems

3. Battle Management/Command, Control, Communication, Computers & Intelligence (BM/C4I) System

The primary goal of BM/C4I is "to provide the warfighter with an integrated BMD capability by building-in the interoperability and flexibility to satisfy a wide range of threats and scenarios." [BMDOLINK] BM/C4I is one of the most important systems of the BMD and is essential in order to take advantage of the full capabilities of the core BMD weapons systems. "Successful BM/C4I increases the time available to engage hostile missiles, increases the effective allocation of interceptors, and reduces the potential for attacking missiles to penetrate U.S. defenses." [BMDOLINK]

According to BMDO, "effective BM/C4I decreases the number of interceptors required by improving weapon system fire distribution and coordination and through sensor fusion." [BMDOLINK] It provides multiple information paths between sensors, shooters, and control locations to combat sensor outages and jamming. BM/C4I weapon cueing information also increases battlespace and depth of fire, improves defense against long-range threats, and increases the defended area. BM/C4I also "supports passive defense measures by providing greater early warning and faster reaction times." [BMDOLINK]

The BM/C4I architecture for TMD is built upon the existing C2 structure for Theater Air Defense (TAD) and adds the communications linking TMD C2 nodes, weapons, and sensors, and the TMD interfaces to intelligence systems and other supporting capabilities. [BMDOLINK] Interoperability in BM/C4I is essential for successful joint TMD operations.

C. CHALLENGES

All ballistic missile defenses are vulnerable to countermeasures. Despite decades of research, dealing with countermeasures remains a key unsolved problem facing missile defenses. It is easier for the attacker to deploy effective countermeasures against defenses than it is for the defense to respond to such countermeasures. [LEWI97]

Effective countermeasures can be cheap and use simple technology – much simpler than the technology required to build long-range missiles. One method the attacker can use is to overwhelm the defense by making the warhead hard to detect, leaving the defense without enough time to intercept it. The attacker can also use methods to prevent the defense from identifying the true warhead. If the United States deploys a national missile defense, it must expect that any developing country that would build or buy long-range missiles to deliver an attack would also make sure these missiles had countermeasures to penetrate the defense. [LEWI97]

Building an effective defense against long-range missiles is difficult even without the use of countermeasures. First, the ground-based radar or satellite-based sensor must detect and track the attacking warhead early enough for the interceptor to reach the warhead. Second, the defense must accurately calculate the projected intercept point and launch an interceptor toward it. Third, the infrared sensor on the interceptor must detect the warhead far enough away to give the interceptor time to maneuver. Finally, the interceptor must maneuver accurately enough to hit the warhead, which is a small object, at a closing speed of greater than 10 kilometers per second (22,000 miles per hour). [LEWI97]

The attacker does not need to do much to make intercepts all but impossible. To defeat a defense, the attacker needs for only one countermeasure to work. For a defense to be reliably effective, it must work against all countermeasures the attacker might use and must work the first time it encounters them. Many countermeasure techniques are available for the attacker to use. Three countermeasure examples are:

1) The attacker can overwhelm the defense.
2) The attacker can make the warhead hard to detect, leaving the defense without enough time to intercept.

3) The attacker can prevent the defense from identifying the true warhead. [LEWI97]

1. Attacker Can Overwhelm the Defense

Chemical and biological warheads can be divided into many small parts called submunitions that can be released early in flight, just after the booster stops thrusting. This creates so many reentering targets that it overwhelms the defense and would therefore defeat any midcourse or terminal defense. Dividing the warhead into submunitions is also beneficial to the attacker because it distributes the chemical or biological agent more efficiently over the target area. The intelligence community has stated that they believe North Korea will be able to deploy submunitions, and that this technology could be available on the world market by 2000. [LEWI97]

2. Attacker Can Make the Warhead Hard to Detect

The infrared sensor on the interceptor, which guides it to the final intercept, detects the heat emitted by the warhead. Cooling the surface of the warhead thus makes it more difficult to detect. A small amount of liquid nitrogen in a thin shroud surrounding the warhead could cool the surface enough to reduce the distance at which the infrared sensor could detect the warhead by 10,000 times. The interceptor would have only a few thousandths of a second to react, in which time it could not maneuver enough to have any chance of intercepting a warhead traveling at 7,000 meters per second. [LEWI97]

Such cooling would also make the warhead much less visible to the infrared detectors on satellite-based sensors such as the planned Space and Missile Tracking System, giving the defense less time to work. Similarly, the warhead can be made more difficult to detect by radar by reducing its radar cross-section using simple techniques such as adding a sharp nose, curving its back end, and covering it with radar-absorbing material. [LEWI97]

3. Attacker Can Prevent the Defense from Identifying the True Warhead

Above the atmosphere, where long-range missiles would be intercepted, objects of different weights and shapes travel at the same speed and follow the same path. This allows a missile to carry a large number of lightweight decoys to confuse the defense. These decoys do not need to be aerodynamic and need not even look like the warhead since the warhead could also be disguised. Such decoys would force the defense either to launch interceptors at all the targets or to wait until the atmosphere strips away the lightweight objects, by which time it could be too late to launch interceptors against the warhead. [LEWI97]

A simple and effective countermeasure is to place the warhead in a metalized mylar balloon and release it within a large cloud of empty balloons. Each of these targets would move at the same speed and could not be distinguished by the missile defense radar. Adding a small heater to each balloon to heat each one by a different amount would prevent infrared sensors from detecting the real warhead. If desired, the attacker could also add a small vibrator to the balloons to mask any small motions the warhead might cause. The lightweight balloons would be stripped away by the atmosphere late in flight, but by that time they would already have done their job. [LEWI97]

4. Summary

Given today's technology, the United States can certainly build a defense that could destroy one or a few ICBM warheads under controlled conditions, where the characteristics of the target warhead are known to the defense and no serious effort is made to defeat the defense. However, the real question is whether the defense will be operationally effective; that is, will the defense be effective in the real world, where the characteristics of the attack would not be completely known in advance and where the attacker would take steps to defeat the defense? The key issue in determining operational effectiveness will be how well the system can deal with countermeasures intended to defeat the system. [LEWI97]

There are many countermeasures that are much easier to build and deploy than is either an ICBM or a nuclear warhead small enough to be delivered on an ICBM. Any country that has developed and deployed nuclear-armed ICBMs has clearly chosen to devote enormous resources to acquiring this capability. Thus, the United States must assume that any country with the technical capability and motivation to deploy an ICBM to attack or threaten the United States will also have the technical capability and motivation to deploy countermeasures to the U.S. BMD system. [LEWI97]

THIS PAGE INTENTIONALLY LEFT BLANK

V. BALLISTIC MISSILE DEFENSE SYSTEM CASE STUDY

A. VIEWPOINTS

This case study is based on the Ballistic Missile Defense (BMD) system. We will demonstrate how the RM-ODP viewpoints can be used to aid in the definition of policy, requirements, and specifications of the BMD system. The BMD system is a good candidate to model using viewpoints because it is a complex, distributed, heterogeneous system comprised of the TMD and NMD subsystems. The TMD system is a multi-tiered family of systems further divided up into lower-tier, upper-tier, and boost-phase subsystems. There are currently three lower-tier subsystems being developed – PAC-3, Navy Area Defense, and MEADS. The two upper-tier subsystems being developed are THAAD and Navy Theater Wide. The only boost-phase subsystem being developed is the Airborne Laser system. Figure 3 depicts the hierarchy of the BMD systems under development.

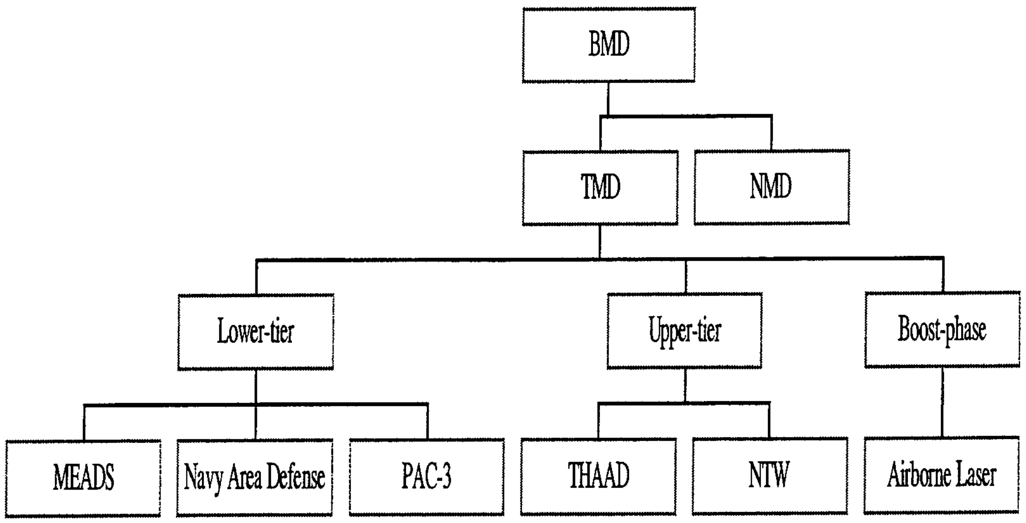

Figure 3. Hierarchy of BMD System

Ballistic Missile Defense involves many challenges. One of these challenges is the ability to hit a moving target. The use of countermeasures by an attacker makes it more difficult to hit the target. Another major challenge facing BMD is interoperability amongst all the systems. The RM-ODP viewpoints can be used to aid in modeling the BMD system to help address some of these challenges.

The first step in modeling the BMD system with RM-ODP is to develop a reference model. The reference model provides the "big picture" that organizes the pieces of an ODP system into a coherent whole. A reference architecture may be derived from the reference model. The reference architecture is a starting point from which detailed specifications can be built.

To construct the reference model of the BMD system, one must look at the "big picture" of the steps required to hit a moving target. First, DSP satellites using infrared sensors would detect an attacking missile. This data is fed into the BM/C4I network. The next step is to calculate the projected intercept point and launch an interceptor toward it. After the boost phase of the attacking missile, sensors in early warning radars are used to track the missile and discriminate the real warhead from decoys. The early warning radars are also used to guide the interceptor. The infrared sensor on the interceptor must detect the warhead far enough away to give the interceptor time to maneuver, and the interceptor must maneuver accurately enough to hit the warhead. Figure 4 illustrates the steps required to hit a target.

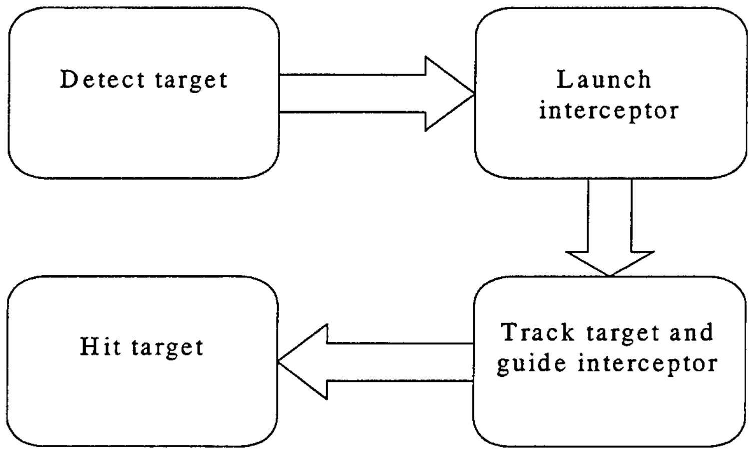

Figure 4. Steps Required to Hit Moving Target

Once the reference model has been identified, the viewpoints can be explored. RM-ODP addresses the problems faced by many organizations today, especially DoD, in choosing the right architecture. This case study uses viewpoints to model the BMD system, where each viewpoint encapsulates part of the requirements for the system. The requirements engineering process for a large, complex system such as BMD involves the participation of many developers with different specialties and roles (such as contractor, project manager, electrical engineering, safety engineering, etc.). Each development team can hold different views of the system. Different contractors and military services are additionally building each of the subsystems of the TMD system. Specifying the BMD system using the RM-ODP viewpoints can allow for an otherwise large and complex specification to be separated into manageable pieces with each focused on the issues relevant to each member of the development team. For example, the systems analyst works with the information specification while the systems programmer is concerned with the engineering viewpoint. The following sections illustrate how the five different RM-ODP viewpoints may aid in modeling the BMD system.

B. ENTERPRISE VIEWPOINT

The enterprise viewpoint focuses on the purpose, scope, and policies for the system. There are many policies to take into consideration when building and deploying such a wide range of ballistic missile defense systems. The success or failure of this system will depend on the technology used in the defenses and the tactics and technologies used in missile attacks. The type of policy that is modeled in this case study is interoperability of all the BMD subsystems. Interoperability is an ongoing challenge for the U.S. military. With the military becoming more joint, interoperability has become an even more important issue.

To achieve interoperability, the policy of interoperability must be translated into requirements and specifications. A BMD system is a real-time, distributed system, so the issues involved are complex. This distributed operating system is a collection of heterogeneous systems connected via a network that presents a global unified view of the system and its resources. It is a tightly coupled system that requires all participants to run their own copy of the same distributed operating system.

In the enterprise viewpoint, policies can be defined in terms of objects, communities, and the roles of the objects within the communities. Within the BMD community are objects, which are either passive or active. Some examples of active objects may be the contractors, systems engineers, program managers, military services, etc. Examples of passive objects could be the radar systems, the interceptors, BM/C4I network, and DSP satellites.

The communities are groupings of objects intended to achieve some purpose. The BMD system can be expressed in terms of a structure of communities. The BMD system as a whole is the community which is comprised of two main subcommunities: TMD and NMD. These subcommunities are further broken down into more subcommunities. Figure 5 is an illustration of the TMD subcommunity portion of the BMD community, which consists of three additional subcommunities.

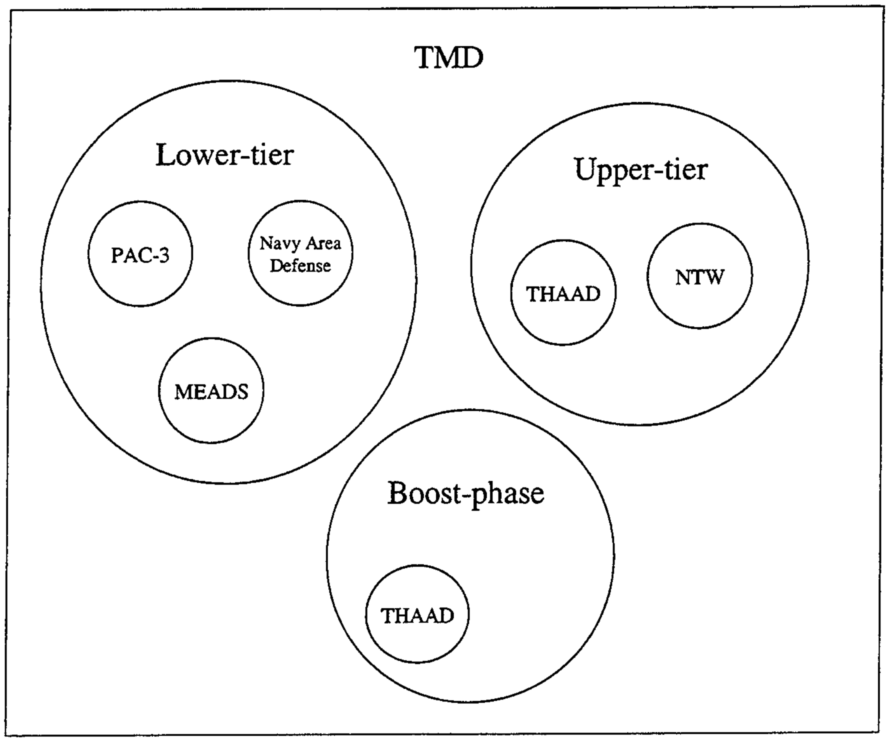

Figure 5. TMD Subcommunity

The roles of the objects, whether passive or active, are expressed in terms of policies. These policies dictate the behavior of the objects within the community. These policies have varying scope and may apply across the community, to a role, or a single action type. The policies can be implicit or explicit. An interoperability policy is an explicit policy. Another explicit policy in the BMD system is a security policy.

When using the enterprise viewpoint, we maintain the highest level of abstraction. When constructing the enterprise viewpoint, it is necessary to view the BMD community as a combination of subcommunities (see Figures 3 and 4). Each of the objects within the subcommunities (TMD and NMD) has specific roles that they must perform. The objects within the three subcommunities of the TMD subcommunity also have specific roles.

Figure 6 is a simplified example of the enterprise viewpoint showing the interoperability policy domain of the BMD system or community. To simplify the figure, only the TMD and NMD subcommunities are modeled.

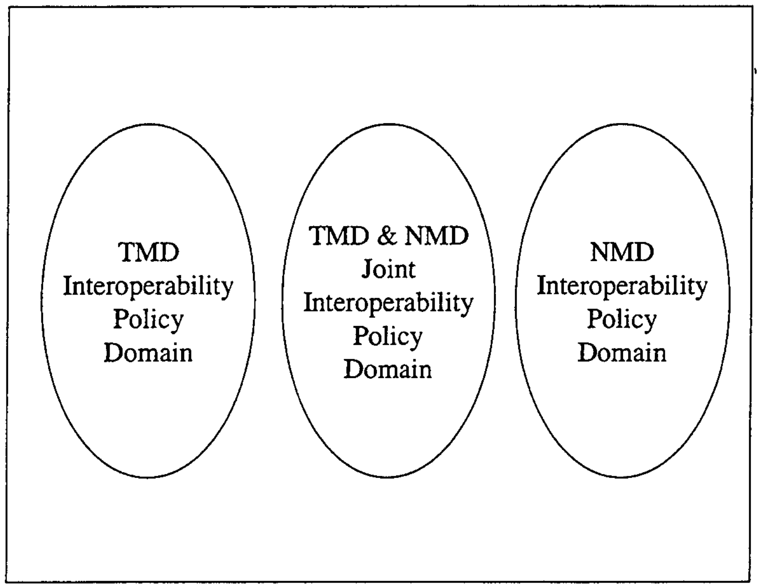
TMD Interoperability Policy Domain
TMD & NMD Joint Interoperability Policy Domain
NMD Interoperability Policy Domain

Figure 6. Enterprise Viewpoint

Figure 6 shows three interoperability policy domains: 1) TMD 2) NMD, and 3) Joint TMD & NMD. Within the TMD interoperability policy domain are interoperability requirements between and within each of the subcommunities of TMD. NMD has interoperability requirements within its subcommunity also. The Joint TMD and NMD interoperability policy domain define interoperability policies between the TMD and NMD domains. An example of a portion of the joint interoperability policy could be the following: the BMD system must be interoperable amongst all of the TMD and NMD subcommunities. A robust interoperable C4I system (BM/C4I) is required to integrate the

BMD subcommunities and provide commanders with the facilities, communications, automated decision aides, and information they need to effectively plan and execute BMD operations.

C. INFORMATION VIEWPOINT

The information viewpoint focuses on the semantics of information and information processing. The information viewpoint is used to describe the information required by the BMD system through the use of schemas, which describe the state and structure of an object. This viewpoint can be used to further decompose the BMD system.

Figure 7 shows an example configuration of RM-ODP objects in an information viewpoint that is appropriate to the policies in the enterprise viewpoint. It is a simplified example of the information viewpoint showing the information supporting the BMD community. Once again only the TMD and NMD subcommunities are modeled. The principle interoperability feature that must be represented is the fact that each of the sets of information entities is isolated from the others and that information only needs to be interoperable at certain levels within the subcommunities.

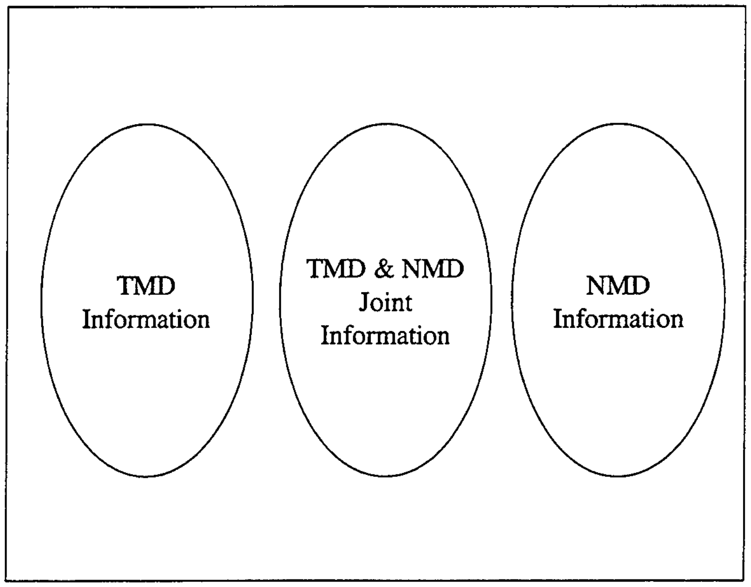

Figure 7. Information Viewpoint

Within the TMD information domain are interoperability requirements both between and within each of the subcommunities of TMD. Each subcommunity within the TMD information domain contains its own version of a radar system, launch system, and interceptor, and thus has different information needs. NMD has separate information requirements within its subcommunity also. The Joint TMD and NMD information domain would define information that is shared between the two subcommunities, and the systems that share this information need to be interoperable.

An example of a TMD interoperability policy domain could include interoperability between the TMD and NDM subcommunities. An example of a portion of the joint interoperability policy is the C4I system required to support the BMD system. A robust interoperable C4I system (BM/C4I) is required to integrate the BMD subcommunities (TMD and NMD) and provide commanders the facilities, communications, automated decision aides, and information they need to effectively plan and execute BMD operations.

D. COMPUTATIONAL VIEWPOINT

The computational viewpoint focuses on the functional decomposition of the system into objects, which interact at interfaces. RM-ODP's computational viewpoint is object-based. The objects encapsulate data and methods, offer interfaces for interaction with other objects, and offer multiple interfaces. The computational viewpoint decomposes the information viewpoint down to a more detailed level.

A computational specification further defines the objects within an ODP system, the activities within those objects, and the interactions that occur among objects. Most objects in a computational specification describe application functionality, and these objects are linked by bindings through which interfaces occur. The binding objects are used to describe complex interactions between objects. Figure 8 is a simplified illustration of the BM/C4I Network object with a lower-tier interface and an upper-tier interface. Both interfaces can be used to destroy a target, but the medium-range targets can only be destroyed through the upper-tier interface and the short-range targets can only be destroyed through the lower-tier interface. Each of the BM/C4I object's interfaces is bound to a TMD object (Navy Area Defense or NTW).

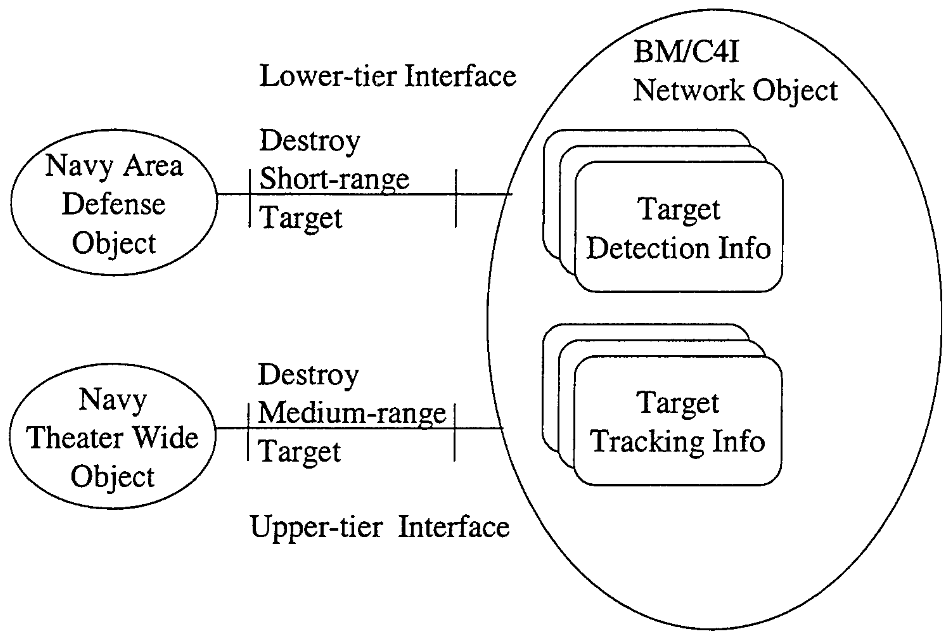

Figure 8. Computational Viewpoint

E. ENGINEERING VIEWPOINT

The engineering viewpoint focuses on the infrastructure required to support distributed interaction between objects in the system. The engineering viewpoint is not concerned with the semantics of the ODP application, except to determine its requirements for distribution and distribution transparency. The computational viewpoint is concerned with when and why objects interact, while the engineering viewpoint is concerned with how they interact.

When modeling interoperability policy, the level of distribution transparency needed must be identified. Of the eight distribution transparencies mentioned in Chapter

2, the failure transparency, location transparency, access transparency, replication transparency, relocation transparency, and persistence transparency should all be considered. Of these transparencies, access transparency and failure transparency are the most important because the BMD system is a heterogeneous real-time system.

Failure transparency masks failure within the distributed system to enhance fault tolerance. If a link or system within the distributed system fails, the entire system should not fail.

Access transparency is also an important transparency for the developer to consider. Access transparency hides the differences in data representation and procedure calling mechanism to enable interworking between heterogeneous computer systems. BMD is comprised of many different hardware platforms, operating systems, and programming languages. RM-ODP does not try to standardize the components of the system or unnecessarily influence the choice of technology.

The fundamental entities described in the engineering viewpoint are objects and channels. Objects in the engineering viewpoint can be divided into two categories – basic engineering objects and infrastructure objects. Basic engineering objects correspond to the objects in the computational viewpoint. An example of an infrastructure object is a protocol object. A channel corresponds to a binding or binding object in the computational viewpoint. A channel provides the communication mechanism and contains or controls the distribution transparencies.

F. TECHNOLOGY VIEWPOINT

The technology viewpoint focuses on the choice of technology for implementation of the system. RM-ODP has very few rules applicable to technology specifications and is difficult to model. This viewpoint further decomposes the system to allow for the choice of the distributed computing model that would be most appropriate. When choosing which distributed computing model to implement, an important issue to consider is the time constraint. The BMD system is a hard real-time system that must satisfy bounded time constraints or suffer severe consequences. The consequences of an undetected missile are severe; if a missile is undetected, it could inflict death and destruction in the targeted area.

The BMD system will also be asynchronous because the events that occur are entirely unpredictable and caused by external sources. One cannot predict when an adversary may launch a missile. Load consideration should also be considered so that a single radar or missile system is not overloaded.

Another important issue to consider when choosing the technology implementation of the BMD system is the network characteristics. There are four network characteristics that can affect the BMD system: 1) network latency, 2) bandwidth versus cost, 3) routing optimization, and 4) micronetwork characteristics. Network latency can cause problems with timing predictability. Distributed real-time applications require constant bandwidth that can be costly. Distributed real-time systems must reside on a network and therefore are subject to the network topology and its affect on distribution and routing. Distributed real-time systems are subject to micronetwork characteristics, such as buffer size, queue sizes, and the packet sizes allowed on the network.

The BM/C4I system is one of the most important systems of the BMD and is essential in order to take advantage of the full capabilities of the core BMD weapons systems. Successful BM/C4I increases the time available to engage hostile missiles, increases the effective allocation of interceptors, and reduces the potential for attacking missiles to penetrate U.S. defenses.

G. SUMMARY

The BMD case study presented only a simplistic view of the BMD system. The case study gave examples of how the RM-ODP viewpoints can be used to model BMD at a very high level. The five RM-ODP viewpoints showed aspects of the distributed system at a high level of abstraction chosen by the developers of the system. Through the use of RM-ODP, the various members of the development team can combine and coordinate together to generate specifications for the system. The developers can combine and coordinate efforts by composing the various views; they also make transformations between levels. When transformations are made between different viewpoints, though, the developers need to ensure that the views are consistent. Consistency is a necessity between levels as well as across viewpoints. Another important feature of RM-ODP is that the viewpoints can be reused at the various levels of abstraction.

VI. CONCLUSIONS AND FUTURE WORK

A. OBSERVATIONS

In this thesis we defined a real operational problem of developing the complex distributed system known as BMD. RM-ODP and viewpoints were defined, and the operational problem was addressed using a viewpoints framework. The case study showed how the viewpoints could be applied at a very abstract level. This thesis demonstrated that the RM-ODP method may be useful for DoD in developing distributed systems.

We hypothesized that the policy, requirements, and specifications could be dealt with at the enterprise level in a very abstract manner. The original hypothesis that the enterprise level would be the level to address interoperability was incorrect. Multiple views must be used to address interoperability policy because issues of interoperability do not tend to surface until you get down to the lower levels. At the conclusion of the case study, it was realized that the system must be viewed down at the technology and engineering levels to make any firm conclusions regarding interoperability.

Interoperability policy can still be implemented at the enterprise level though because it can be made explicit in the policy and requirements that interoperability is required at certain levels within and between various components. Viewpoints actually help to decompose the problem into manageable pieces at each level. The members of the development team, each with different roles, are able to view the system at various levels of abstraction. A systems analyst would not be working at the engineering level but at a higher level in the hierarchy of views. The engineering level is concerned with hardware platforms and how they interoperate. Viewpoints support the division of labor and the idea of hierarchy of tasks.

The viewpoints framework can aid in detecting and managing any inconsistencies that may arise in the requirements and specifications of each of the BMD systems. In the BMD system, the BM/C4I system will link the subsystems to each other to support a multi-tiered, highly effective, interoperable BMD architecture.

Inconsistencies provide clues about missing information. A reason for dealing with inconsistencies is to provide a more robust specification. Viewpoints can be used as the cornerstone of effective inconsistency management.

The main goal in applying RM-ODP during system development is to detect as many conflicts as possible at the time of specification, rather than leaving them to be detected at runtime. Conflicts that are not detected until runtime can endanger the lives of the military and many civilians. The developers can modify the policies to remove conflicts in requirements and specifications.

Viewpoints will help determine where interoperability is necessary and sufficient, and where interoperability should be optional or does not make sense. Interoperability in some systems may only be at the communications level where you are passing data back and forth to respond to a threat and each system goes about addressing that threat however it is best suited to do so. Two systems could work in a coordinated manner.

RM-ODP can be used to help us address these issues. It brings out the fact that we need interoperability. One system may use a certain type of radio for communications. At the technical level we may discover that we have two different types of radios that are not compatible. What would that tell you? You may need to rethink your system and go back up to a higher level where your requirements may state that you must use interoperable communications, and then as we come back down we may see the discrepancies.

B. FUTURE WORK

It is important to realize that RM-ODP is a framework that permits us to focus on different viewpoints of a system. The case study showed how the viewpoints could be applied at a very abstract level and how RM-ODP may be useful for DoD in developing distributed systems. Future research topics could include the detailed application of the enterprise, information, computational, engineering, and technology viewpoints to other information systems being developed. Each of the five viewpoints could also be individually expanded on to model different development views of the system. The area of transformations between the different levels of viewpoints and consistency could be also explored. When transformations are made between different viewpoints, the developers need to ensure that the views are consistent. Consistency is a necessity between levels as well as across viewpoints. Another area of research could be the idea of viewpoint reuse. An important feature of RM-ODP is that the viewpoints can be reused at the various levels of abstraction.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A: ACRONYMS

ABL	Airborne Laser
BM/C4I	Battle Management, Command, Control, Communications, Computers, and Intelligence
BMD	Ballistic Missile Defense
BMDO	Ballistic Missile Defense Organization
C-1	Capability-1
C-2	Capability-2
C-3	Capability-3
C2	Command and Control
C4I	Command, Control, Communications, Computers, and Intelligence
CORE	Controlled Requirements Expression
DoD	Department of Defense
DSP	Defense Support Program
ECS	Engagement Control Station
EKV	Exo-atmospheric Kill Vehicle
FoS	Family of Systems
GBI	Ground Based Interceptor
GEM	Guidance Enhanced Missile
ICBM	Intercontinental Ballistic Missile
IEEE	Institute of Electrical and Electronics Engineers
IFICS	In-Flight Interceptor Communications Systems
ISO	International Standards Organization
ITU-T	International Telecommunication Union – Telecommunications Standardization Sector
LEAP	Lightweight Exo-Atmospheric Projectile
MEADS	Medium-range Extended Air Defense System
NMD	National Missile Defense
NTW	Navy Theater Wide
ODP	Open Distributed Processing
PAC-2	Patriot Advanced Capability-2
PAC-3	Patriot Advanced Capability-3
RM-ODP	Reference Model of Open Distributed Processing
SADT	Structured Analysis and Design Technique
SAM	Surface-to-Air Missile
SBIRS-high	Space-Based Infrared System-high-earth orbit
SBIRS-low	Space-Based Infrared System-low-earth orbit
SLBM	Submarine Launched Ballistic Missile
SM-2	STANDARD Missile-2
SRD	Structured Requirements Definition
TAD	Theater Air Defense
TBM	Theater Ballistic Missile
TBMD	Theater Ballistic Missile Defense
TD-1	Taepo Dong-1
TD-2	Taepo Dong-2
THAAD	Theater High Altitude Area Defense
TMD	Theater Missile Defense
UEWR	Upgraded Early Warning Radar
VOA	Viewpoint-Oriented Analysis

VOSD	Viewpoints-Oriented Software Development
VWPL	Viewpoint Language
XBR	X-Band Radar

LIST OF REFERENCES

[BMDOLINK] BMDOLINK. Ballistic Missile Defense Organization homepage. http://www.acq.osd.mil/bmdo/bmdolink/html.

[BMDO00A] "National Missile Defense Program," BMDO Fact Sheet JN-00-05, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, January 2000.

[BMDO99A] "Ballistic Missile Defense – The Core Programs," BMDO Fact Sheet AQ-99-01, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, February 1999.

[BMDO99B] "Navy Area Ballistic Missile Defense Program," BMDO Fact Sheet AQ-99-02, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, February 1999.

[BMDO99C] "Navy Theater Wide Ballistic Missile Defense Program," BMDO Fact Sheet AQ-99-03, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, February 1999.

[BMDO99D] "PATRIOT Advanced Capability-3," BMDO Fact Sheet AQ-99-04, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, February 1999.

[BMDO99E] "Theater High Altitude Area Defense System," BMDO Fact Sheet AQ-99-05, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, July 1999.

[BMDO99F] "Medium Extended Air Defense System," BMDO Fact Sheet AQ-99-11, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, February 1999.

[BMDO99G] "The Family of Systems Concept," BMDO Fact Sheet AQ-99-16, http://www.acq.osd.mil/bmdo/bmdolink/html/factsheet.html, March 1999.

[EAST96] Easterbrook, Steve and Bashar Nuseibeh. "Using ViewPoints for Inconsistency Management." *Software Engineering Journal*, Volume 11 1, January 1996, Pages 31-43.

[ENTE99] "Open Distributed Computing," JTC1/SC7, http://enterprise.shl.com

[FEUS99] Feustel, Edward A. "RM-ODP and Security: A Coalition Example." Workshop on Education for Computer Security, Asilomar, CA, 1999.

[FINK89] Finkelstein, Anthony and Hugo Fuks. "A Cooperative Framework for Software Engineering." In Proceedings of the Twenty-second Annual Hawaii International Conference on System Sciences, 1989, Pages 189-199.

[FINK94] Finkelstein, Anthony, Dov Gabbay, Anthony Hunter, Jeff Kramer, and Bashar Nuseibeh. "Inconsistency Handling in Multiperspective Specifications." *IEEE Transactions on Software Engineering, Volume: 20 8*, August 1994, Pages 569-578.

[HERR97] Herring, Charles, Zoran Milosevic, and Simon Kaplan. "Selecting Distributed Object Technologies in the Presence of Uncertainty: an Experience Report on C4I enterprises modelling." In Proceedings of the First International Enterprise Distributed Object Computing Workshop, 1997, Pages 245-256.

[ISO195] ISO/IEC 10746-1. "International Standard 10746-1, ITU-T Recommendation X.901, Reference Model of Open Distributed Processing – Part 1: Overview," May 1995.

[ISO295] ISO/IEC 10746-2. "International Standard 10746-2, ITU-T Recommendation X.902, Open Distributed Processing - Reference Model - Part 2: Foundations," 1995.

[ISO395] ISO/IEC 10746-3. "International Standard 10746-3, ITU-T Recommendation X.903, Open Distributed Processing - Reference Model - Part 3: Architecture," January 1995.

[ISO495] ISO/IEC 10746-4. "International Standard 10746-4, ITU-T Recommendation X.904, Basic Reference Model of Open Distributed Processing - Part 4: Architectural Semantics," 1995.

[KOTO92] Kotonya, Gerald and Ian Sommerville. "Viewpoints for Requirements Definition." *Software Engineering Journal*, Volume 7 6, November 1992, Pages 375-387.

[LEWI97] Lewis, George N. and Theodore A. Postol. "Future Challenges to Ballistic Missile Defense." *IEEE Spectrum, Volume: 34 9*, September 1997, Pages 60-68.

[LUPU97] Lupu, E. and M. Sloman. "Conflicts in Policy-base Distributed Systems Management." In Proceedings of the Fifth IEEE/IFIP International Symposium on Integrated Network Management, May 1997, Pages 1-29.

[MICH99] Michael, J. Bret. "A Look at RM-ODP from an Information Assurance Perspective." Workshop on Domain Modeling, Institute for Defense Analyses, Alexandria, VA, 1999.

[MOSH97] Mosher, David E. "The Grand Plans." *IEEE Spectrum, Volume: 34 9*, September 1997, Pages 28-39.

[NUSE92] Nuseibeh, Bashar and Anthony Finkelstein. "ViewPoints: A Vehicle for Method and Tool Integration." In Proceedings of the Fifth International Workshop on Computer-Aided Software Engineering, 1992, Pages 50-60.

[NUSE93] Nuseibeh, Bashar, Jeff Kramer, and Anthony Finkelstein. "Expressing the Relationships Between Multiple Views in Requirements Specification." In Proceedings of the 15th International Conference on Software Engineering, 1993, Pages 187-196.

[NUSE94] Nuseibeh, Bashar, Jeff Kramer, and Anthony Finkelstein. "A Framework for Expressing the Relationships Between Multiple Views in Requirements Specification." *IEEE Transactions on Software Engineering, Volume: 20 8*, August 1994, Pages 760-773.

[SIBL92] Sibley, Edgar H., James Bret Michael, and Richard L. Wexelblat. "Use of an Experimental Policy Workbench: Description and Preliminary Results." *Database Security, V: Status and Prospects.* C.E. Landwehr and S. Jajodia, eds. Elsevier Science Publishers, Amsterdam, 1992, Pages 47-76.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center...2
   8725 John J. Kingman Road, Suite 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library...2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, CA 93943-5101

3. Dean Dan Boger 1
   Chair, C3 Academic Group
   Naval Postgraduate School
   Monterey, CA 93943

4. Professor J. Bret Michael 2
   Department of Computer Science,
   Code CS/M
   Naval Postgraduate School
   Monterey, CA 93943-5118

5. Professor William G. Kemple 1
   Command, Control, Communications Academic Group
   Code CC/Ke
   Naval Postgraduate School
   Monterey, CA 93943

6. LT Sheila A. Smith..2
   105 Greentree Court
   O'Fallon, IL 62269

7. Mr. Arison 1
   JCS/J6
   Pentagon
   Washington, D.C. 20350-2000

8. Capt Maslowsky 1
CNO/N62
Pentagon, Washington D.C. 20350-2000

9. Lt Col Ziegenfuss 1
HQMC C4I Plans and Policy Division
2 Navy Annex
Wash D.C. 20380-1775

10. Dr. Dennis Fife 1
Computer and Software Engineering Division
Institute for Defense Analysis
1801 N. Beauregard St.
Alexandria, VA 22311-1772

11. Dr. Edward A. Schneider 1
Computer and Software Engineering Division
Institute for Defense Analysis
1801 N. Beauregard St.
Alexandria, VA 22311-1772